Copyright ⓒ 2022 KSII

A Secure Subscription-Push Service Scheme Based on Blockchain and Edge Computing for IoT

Yinjuan Deng[1, 2], Shangping Wang[1*,3], Qian Zhang[4] and Duo Zhang[1]
[1]Shool of Automation and Information Engineering, Xi'an University of Technology
Xi'an710054, China
[e-mail: zhangduo029@163.com]
[2]School of Mathematics and Information Science, Baoji University of Arts and Sciences
Baoji721013, China
[e-mail:dmy1104@126.com]
[3]Shool of Science, Xi'an University of Technology
[e-mail:spwang@mail.xaut.edu.cn]
[4]Shool of Computer Science and Engineering, Xi'an University of Technology
[e-mail: qianz95119@126.com]
[*]Corresponding author: Shangping Wang

Received May 24, 2021; revised September 14, 2021; revised December 8, 2021; accepted January 24, 2022; published February 28, 2022

Abstract

As everything linking to the internet, people can subscribe to various services from a service provider to facilitate their lives through the Internet of Things (IoT). An obligatory thing for the service provider is that they should push the service data safely and timely to multiple IoT terminal devices regularly after the IoT devices accomplishing the service subscription. In order to control the service message received by the legal devices as while as keep the confidentiality of the data, the public key encryption algorithm is utilized. While the existing public encryption algorithms for push service are too complicated for IoT devices, and almost of the current subscription schemes based on push mode are relying on centralized organization which may suffer from centralized entity corruption or single point of failure. To address these issues, we design a secure subscription-push service scheme based on blockchain and edge computing in this article, which is decentralized with secure architecture for the subscription and push of service. Furthermore, inspired by broadcast encryption and multicast encryption, a new encryption algorithm is designed to manage the permissions of IoT devices together with smart contract, and to protect the confidentiality of push messages, which is suitable for IoT devices. The edge computing nodes, in the new system architecture, maintain the blockchain to ensure the impartiality and traceability of service subscriptions and push messages, meanwhile undertake some calculations for IoT devices with limited computing power. The legalities of subscription services are guaranteed by verifying subscription tags on the smart contract. Lastly, the analysis indicates that the scheme is reliable, and the proposed encryption algorithm is safe and efficient.

This research is supported by Shaanxi Province Key Research and Development Program, No. 2020GY-006, and the Project of ShaanXi Education Department, No. 19JK0042

Keywords: Blockchain, Edge Computing, Encryption Algorithm, Internet of Things Devices, Subscription-Push Service.

1. Introduction

Internet of Things is one of growing technologies [1], the connection of machine to people makes our life style change qualitatively. IoT devices may need to subscribe to some services, such as weather, stock, traffic, radio and television data etc., to make people's life more intelligent [2]. The requirement of the service is real-time and long-term. In the distributed Internet of Things environment, push and pull are two main ways for information transmission. Pull refers to the fact that the users request specific data from a service provider actively. While the push is on the contrary, service provider takes the initiative and sends messages to users without request from them. In our scenario, IoT devices need to get service message on the same subject from service provider frequently. If using pull mode, IoT devices need to send polling requests constantly, which waste the network broadband and processor time significantly. So, the push mode is adopted naturally.

The service provider is for commercial interest, as long as the IoT devices have paid for subscribing to a certain theme service, the service provider needs to send service data to them regularly, and those without payment cannot get the service contents. So, the service provider cares about the access authorities of the receivers and the confidentiality of the transmission content when pushing the same service content to multiple devices simultaneously through an open wireless network.

As IoT devices are the subscribers, with limited computing power, so the designed scheme should consider the computation cost required for IoT devices when decrypting the confidential message, as well as the problems about economic issues and service revocation after subscription.

As far as we know, such issues as confidential transmission, the recipients' access rights and possible economic disputes etc. mentioned above have not been taken into account comprehensively in the existing architectures or schemes. Publish/subscribe system has the function of subscription and push information [3], and realizes the access rights of the receivers naturally. However, the system is aimed at complex information sharing scenarios, needs middleware to match and filter the information, and takes little consideration of security. In our design goals, IoT devices can subscribe to services such as weather, health information, etc. directly, and no middleware is required. Broadcast encryption and multi-receiver encryption can be used for secure one-to-many communication [4], as while as achieve push mode, what's more, the access control realized by using secret keys skillfully. These encryption methods meet service provider's requirements nicely. Unfortunately, the decryption burden is too heavy for the IoT devices and can't realize flexible join and revocation of users in most broadcast encryption and multi-receiver encryption algorithms existing currently. And the economic issues and key distribution in subscription are generally solved by centralized server, this implies that the server must always be online and credible, and the system will fail if the server is compromised.

Considering above requirements from the service provider and IoT devices, the designed scheme should have following properties and functions:

(1) Confidentiality of pushed service message

(2) Control the access rights of IoT devices

(3) The system can realize decentralized payment and subscription verification, enable flexible subscription and revocation.

(4) The proposed algorithm has relatively small computation cost for IoT devices, it is secure and has high efficiency.

To address the questions above simultaneously, a secure subscription-push service scheme suitable for IoT devices is proposed in this paper, in which the framework is benefiting from the inspiration of the publish/subscribe system as while as blockchain. The access control and confidentiality solution benefits from broadcast encryption [5] and multi-receiver encryption [6].

The main contributions of our works can be summarized as follows:

(1) A decentralized subscription-push service scheme is proposed. The subscription and verification of push services are all carried out on the blockchain. By deploying a smart contract for subscription-push service, the fairness, immutability, and traceability of service are guaranteed, and possible service disputes can be resolved.

(2) An encryption algorithm is tailored for subscription-push service, so that the service provider pushes service messages for multiple IoT devices securely. And the algorithm controls the access authorities of IoT devices with smart contract jointly.

(3) Edge computing is leveraged in the subscription-push service scheme. One function of the edge computing nodes is to maintain the blockchain, and the other is as an agent of IoT devices to preprocess the ciphertext, so that the IoT devices have very little decryption computation, which is suitable for them.

(4) The subscription and revocation of service can be achieved flexibly in our designed subscription-push service scheme through smart contract which is tamper-proof and traceable.

2. Related work

2.1 The solutions of push message for IoT

Subscription and push service has been implemented in several systems and protocols, such as publish/subscribe system [7], broadcast message, MQTT protocol [8], etc. Just like the service deployment and discovery platform designed for the Internet of Things in [9], the authors applied mobile edge computing (MEC) to solve the problem of service providers providing timely and efficient services for IoT devices subtly. A gateway is organized in every area according to a three-tiered hierarchical MEC network topology to reduce computing overhead at the centralized controller. Zhu et al. [10] designed a time-based access control mechanism for a specific subscription system, in which the use of cryptographic algorithm enables the legitimate receivers to get the push information as while as guarantees the security of the information. Pan et al. [2] proposed an intelligent push message service solution for IoT devices. In their scheme, the matching algorithm is mainly studied to realize intelligent push services, and some restrictions are designed for IoT devices when receiving messages. Another subscription protocol that supports the push model is the MQTT protocol [8], which is lightweight and designed for the Internet of Things (IoT) environment. Although the message pusher and receivers are decoupled in MQTT, the message pusher cannot know whether there are illegal users stealing the messages when he performing push service. And

the messages are transferred in plain text, which is a great of threat to the security. In order to design a decentralized communication model, in the literature [11], the authors gave a blockchain-based secure communication scheme in which a general security authentication key management solution based on blockchain is designed for the internet of intelligent things (IoIT). And the details of the associated system models and a practical demonstration of the proposed generalized scheme are provided for the blockchain-based IoT communication environment.

The above schemes can implement message push for IoT scenarios. While in literature [2][7][8][9], the confidentiality of data is not considered. In literature [10] the authors achieve the service push based on a specific subscription system, while it is centralized, and may be vulnerable to single point of failure. In [11] authentication and access control are implemented in the decentralized architecture, while no subscription process is involved.

2.2 The solutions of secure push message

One pusher and multiple receivers are involved in our system. Considering the confidentiality, the message pushed should be encrypted, and the access rights of the receivers are determined by the keys. Broadcast encryption, multi-recipient encryption and hierarchical identity-based encryption [12] fulfills these requirements. However, there are some problems when applying these encryption algorithms to our scheme directly, so we'd better to discuss the related work. In the broadcast encryption scheme proposed by B. Dan et al. [13], a receiver has to receive his own private key and the public keys of all users sent by the system in advance, and store these keys locally. It is not realistic for IoT devices to save so many keys, and there is no mention about revocation management of a user in their scheme. After that, the key length of a user was improved in some broadcast encryption schemes [14] [15], however, they are still not applicable to IoT devices. Another drawback is that the system parameters need to be regenerated in those schemes when revoking a user. Lewko et al. [16] proposed a revocable broadcast encryption scheme, the lengths of public key and private key in their scheme are short. However, the length of the ciphertext is related to the number of users revoked. The revocation work can be done on the premise that the identities of the revoked users determined in advance, which makes it a troublesome business. Du et al. [17] proposed an identity-based broadcast encrypted scheme on the basis of identity-based encryption, where the public key is generated by the identity of a user. A defect of the scheme is that the length of the ciphertext is linearly to the number of users, so there is a high requirement on the communication and bandwidth. The scheme allows users satisfied the identity conditions to get the key and decrypt ciphertext. In order to control the receivers of broadcast encryption more flexibly, researchers have proposed attribute multicast encryption schemes [18][19][20], where the flexible revocation is realized. However, the fine-grained access control is their main objective, which makes the calculation is more complicated for IoT device nodes. The hierarchical identity-based encryption and prefix encryption given by Lewko et al. [21] can also implement push service, in which the service provider can send a ciphertext to multiple receivers at the same time as long as the identity vector is a prefix of the given identity, but the decryption computation is a burden for IoT devices. Multi-recipient encryption algorithm, another cryptographic primitive, can also enable one party to send encrypted message and multiple parties to decrypt it. Tseng et al. in [22] gave an efficient ID-based multi-recipient encryption scheme with flexible revocation. In their scheme, a pairs operation is required in encryption process only, but our application scenario pays more attention to the decryption computation. The aforementioned few types of encryption schemes have security risk due to the key disclosure problem, because they assume the keys are generated by a trusted third party. In order to weaken the direct control of the key by a third party, the schemes [23][24][25][26][27]

show certificateless multi-recipient encryption algorithms where user's partial private key is generated by the key generation center (KGC), and the secret value is generated by himself. The two parts are all required when performing decryption operation. Deng's decryption algorithm [25] contains pair operation which does not suit for IoT devices. He et al. [23] proposed a certificateless multi-recipient encryption scheme for mobile devices with small computing power, and decryption is a scalar multiplication operation. The schemes from Pang [26] and Hung [24] also reduce the amount of decryption calculations subtly, but their schemes do not consider the problem of revocation. Peng et al. [27] designed a lightweight multi-receiver signcryption scheme for the Internet of Things, but the scheme did not involve outsourcing decryption. In addition, the certificateless cryptographic system exist some security risks.

In order to consider communication bandwidth and decryption computation, as well as some other indicators in relevant literature, a comparison between several encryption algorithms is given in **Table 1**.

After the summary and analysis of related schemes, it can be seen that the algorithms mentioned in section 2.2 are all realized the confidentiality and access control of push messages. However, some decryption operations are unbearable for IoT devices [19] [20] [21] [25] because of bilinear-pairing operation. And the lengths of ciphertexts in most schemes are related to the number of users, which may lead to a low communication efficiency [23][24][19]. And it may lead to single points of failure because of centralized or certificateless in these encryption algorithms.

Through the analysis of related work in section 2, we found that they are suitable in their own scenarios. Unfortunately, there will be such problems as the differences in the function of some one component in the architecture, lack of revocation functionality, or heavy operations etc. when employing them to solve the difficulties we faced. Until now, no one meet our all requirements at the same time. Inspired by the literature, we will design a new architecture with decentralized feature and tailor an encryption algorithm to match the architecture as well as suit to IoT devices, to achieve secure and efficient subscription-push and cancellation of services, as well as other properties we mentioned in introduction section.

Table 1. Comparison of several encryption algorithms

Schemes	Ciphertext length	Decryption	Outsou -rced	Decentralized and its form	Revoca -tion	λ_c
[20]	$(4n-4l+3)L_p+L_q+\lvert S\rvert$	$3E+2P$	Yes	No	No	$O(n)$
[25]	$2(w+L_p)+(n+1)L_q$	$P+E$	No	Certificateless	No	$O(n)$
[26]	$(3+n)L_p$	$2PM$	No	Certificateless	No	$O(n)$
[23]	$(2n+1)L_q+L_p$	$2PM$	No	Certificateless	No	$O(n)$
[24]	$2nw+L_p+L_q$	$P+PM$	No	Certificateless	No	$O(n)$
[19]	$2rL_p$	$3P+2rE$	No	No	Yes	$O(r)$
[27]	$(2+n)L_q+L_p$	$4PM$	No	Certificateless	No	$O(n)$
Ours	$3L_p$	E	Yes	blockchain	Yes	$O(1)$

Where λ_c is the relationship between ciphertext length and the number of receivers, n represents the number of receivers, P represents a pair operation, E represents an exponential calculation, PM represents a scalar multiplication. Other operations, such as hash and inversion etc. are simple and take little time, we omit them [20]. L_p (L_q) is the length of a

member in a group $Z_p\left(Z_q\right)$, w is the output length of the hash function, r represents the number of revocable users, l indicates the number of groups, $|S|$ represents the number of different identities.

3. Preliminaries

DDH assumption in cryptographic algorithm is utilized to ensure the security of the encryption algorithm. Blockchain, smart contract and edge computing are some basic components in our architecture.

3.1 Deterministic Diffie-Hellman (DDH) problem

Let p is a large prime number, G is a cyclic group with prime order p, g is a generator of G, and $a,b,c \in Z_p^*$. The DDH problem refers to distinguish $\left(g,g^a,g^b,g^{ab}\right)$ and $\left(g,g^a,g^b,g^c\right)$. If $\left|pr\left[A\left(g,g^a,g^b,g^{ab}\right)=1\right]-pr\left[A\left(g,g^a,g^b,g^c\right)=1\right]\right| \le \xi$, then we say the algorithm A solves the DDH problem with advantage ξ ,whose advantage function is defined as $\mathrm{Adv}_G^{DDH}=\xi$.

DDH assumption means that for any polynomial algorithm adversary A to solve the DDH problem with negligible advantages $\mathrm{Adv}_G^{DDH}=\xi$.

3.2 Blockchain and Smart Contract

Blockchain technology [29] is the underlying technology for digital currencies such as bitcoin and ether. It can realize fair trades and no-tampered accounting, information sharing and the design of distributed Internet protocol. Blockchain is a chained list linked by blocks that can store transaction records or data [30]. Once the records and data are on chain, they are no-tampered. With the development of the blockchain architecture, smart contract can be deployed on the Ethereum blockchain [31], which is a digital protocol executed automatically without a trusted third-party to monitor. Smart contract executes smart code through external calls and functions, and generates events, then broadcasts them to all participants [32]. Smart contract, is just a transaction, can be packaged into blocks. In this solution, a smart contract is deployed by service provider to verify some crucial information during subscription and push service.

3.3 Edge computing

Edge computing [33] refers to the technical services that perform computing on the edge of the network. That is on the downstream data of cloud services and the upstream data of the Internet of Things. Smartphone, for example, is the edge between wearable devices and the cloud. Edge computing can handle a large number of data calculations. When edge computing nodes keep good quality, they have lower latency and better broadband than cloud nodes [34]. The edge can perform operations such as computing, offloading, data storage, caching, and processing. The edge computing nodes mainly play two roles in this paper, one is to maintain the blockchain, and the other is working as outsourcing services, undertaking storage and computing for service provider and IoT devices respectively [35].

4. System architecture and attacks model

4.1 Notation description

The notation description shows in **Table 2**.

4.2 System architecture

The participants in this paper include three entities: IoT devices, edge computing nodes, and a service provider. For simplicity, only one service provider is involved in our scheme, he can provide multiple service items.

IoT devices have low computing capability. In cryptographic algorithms, they are suitable for calculating some simple operations, such as hash, exponent operation, inversion and scalar multiplication operation on ECC, etc., while pairs-based operations are unbearable for them [36][37]. They subscribe to some required services from service provider, and receive push service and preprocessing data from the service provider and an edge computing node nearby respectively.

Edge computing nodes are equipped with strong computing power and enough storage capacity. A part of the computing power and storage of an edge computing node works as a blockchain miner node. And the rest are rented by the service provider.

The service provider mainly pushes session keys and service information to IoT devices. A subscription-push service smart contract would be deployed by SP on the blockchain for automated trades. All the service names and charges are sent to the blockchain in advance by service provider.

Table 2. List of Notations

Notation	Description	Notation	Description
U	IoT end devices	SP	Service provider
ECN	Edge computing nodes	S	Services provided by service provider
U_i	The ith IOT end device	ECN_k	The kth edge computing node
S_j	The name of the jth service	Enc	Our encryption algorithm
K	Symmetric encryption key	id_i	The identity of the ith device
Enc_K	Symmetric encryption algorithm with K	$\omega_{i,j}$	The evidence of the ith device subscribe to the jth service
$usk_{i,j}$	Private key of the ith IoT device for the jth service	$psk_{i,j}$	Preprocessing key of the ith IoT device for the jth service

The scheme consists of four steps as follows. The specific flow chart is shown in **Fig. 1**.

Step1. IoT devices subscribe to services based on blockchain.

①An IoT device queries the required service name and the charge on the blockchain, then sends the prepayment of the selected service, its identity and payment time to the smart contract deployed by the service provider. ②After receiving the message of successful prepayment returned by the smart contract, IoT device sends its public key, service name to be subscribed and its identity as subscription messages to the nearest edge computing node for local storage. ③The node that received the subscription information checks whether the IoT device has prepaid for the subscription service on the smart contract. ④If the device has finished the prepayment, the edge computing node sends the subscription information of the IoT device to the service provider. The information includes the service name, some public

parameters processed by the edge computing node for the device, and the identity of the IoT device. At the same time, the subscription information is stored in a subscription information table created for IoT devices by the edge computing node. And the subscription service record in the table is marked as "has verified in the contract". ⑤The service provider sends the subscription tag and relevant information used for preprocessing ciphertext subsequently to the edge computing node for local storage. At the same time, the subscription tag and the identity of the IoT device are sent to the blockchain by SP for subsequent verification. ⑥After receiving the information, the edge computing node stores them in the subscription information table for the IoT device, and notifies the device it has subscribed to the service successfully.

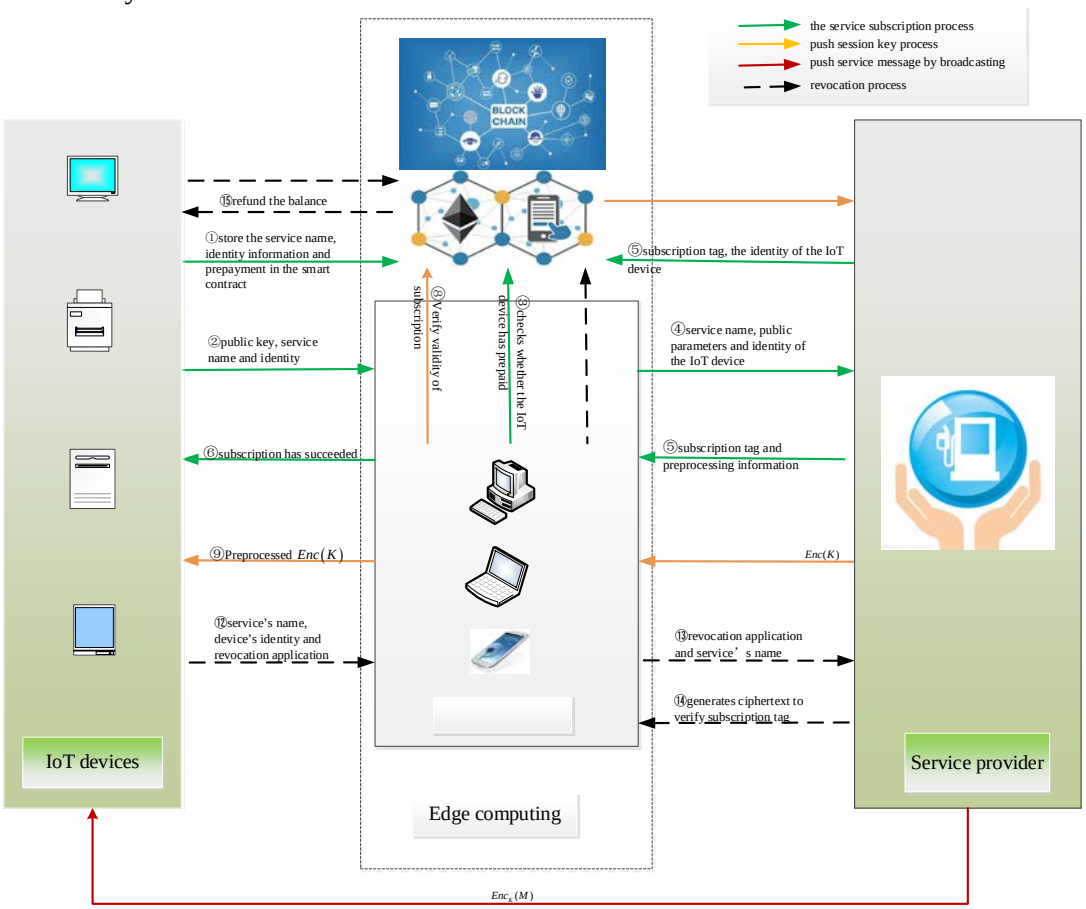

Fig. 1. System architecture and flow chart

Step2. Service provider pushes session key to legitimate IoT devices.

We call a device successfully subscribe to a service is legitimate. When the subscription tag passes the verification, the smart contract will transfer a certain amount of service fee from the account of IoT device to the service provider's account automatically.

⑦The service provider pushes the ciphertext of a session key by broadcasting to all edge computing nodes. ⑧After receiving ciphertext from service provider, every edge computing node verifies whether an IoT device in its subscription information table has subscribed to the service through its subscription tag, and sends identity and subscription tag of the IoT device to the smart contract for confirmation. After the verification is passed, the expense of this time

will be deducted from the device's account and transfer to the service provider's account through the smart contract automatically, otherwise the node verifies the next device in its table. ⑨If the verification is passed, the edge computing nodes preprocess the ciphertext for every subscription device, and sends the preprocessed ciphertext for the device to decrypt again and get the session key.

Step3. SP pushes encrypted service data to IoT devices by broadcasting.

⑩The service message is encrypted with the session key and then pushed by the form of broadcast to the IoT devices by service provider, then the IoT devices with the keys can decrypt ciphertext and get the service message.

Step4. Blockchain-based service revocation.

An IoT device can apply for revoking a service and be refunded the balance.

⑪Service's name and device's identity are sent to the smart contract to apply for revoking a service by an IoT device. ⑫After the revocation is confirmed by the smart contract, the service's name and device's identity are sent to the nearest edge computing node where has applied for subscription by the IoT device. ⑬After receiving the application of revocation, the edge computing node submits it and the service's name to the service provider. ⑭Service provider generates a ciphertext, and sends it to the edge computing node to verify subscription tag. ⑮The edge computing node verifies the subscription tag of the IoT device for this service. If the verification is passed, the subscription tag, identity, and service's name are sent to the smart contract to apply for service revocation. After receiving the request, smart contract checks whether the IoT device has submitted the service revocation application. If so, the remaining fee of the IoT device will be refunded, and the subscription tag recorded by the smart contract will be deleted, otherwise it is an invalid request.

4.3 Threat model and security model

4.3.1 Threat model of malicious edge computing node

1) A malicious edge computing node may not preprocess ciphertext for an IoT device in order to save computing power.

2) A malicious edge computing node may collude with an IoT device and sends preprocessed ciphertext to the device without verification on the smart contract so that the service fees of the device will not be deducted.

3) Malicious edge computing node can eavesdrop on the open channels and get the ciphertext of session key, public key information of the device. It can inquire the information on the blockchain and stores some secret information related to a device. It also can issue subscription process queries adaptively to multiple devices, and obtain the information for pre-decryption given by the service provider. It tries to get the session key using all the information obtained.

4.3.2 Threat model of malicious IoT device

1) Device without subscription may eavesdrop on the open channels to get some information like ciphertext, public key etc. It tries to get a session key.

2) A subscription expired device with an expired session key, can eavesdrop on the open channels and get all the messages on the channels. It can inquiry multiple legitimate IoT devices and goes through the registration and decryption process as training. It wants to impersonate a legitimate IoT device which has not been inquired before to subscript to a service without payment.

4.3.3 Threat model of malicious service provider

The service provider is not honest and trustworthy completely. He may not distribute subscription tag for a certain IoT device.

4.3.4 The security model of proposed encryption algorithm

We give the security model of our encryption algorithm here. The core algorithm of this scheme is pushing session (symmetric) key to IoT devices. The session key is regarded as the plaintext before encryption here, then, our encryption algorithm should be secure under Chosen-Plaintext Attack (CPA). The security is defined by an interactive game between an adversary A and a challenger C. In the game, we assume that the adversary can eavesdrop on all the edge computing nodes, or simulate a legitimate IoT device to ask the processes of subscription and push a key adaptively. And two blank lists will be created to record some inquiries from the adversary, and the respondents are all simulated by the challenger. One is used for recording the inquiries of simulating a legitimate device, called $List1=\left\{G_i, usk_i, F_i, K_i, C_i, C_i'\right\}$. And the other records the inquiries of monitoring an edge computing node, called $List2=\left\{G_j, psk_j, F_j, K_j, C_j, C_j'\right\}$.

In list1, G_i represents the information of the ith device, usk_i is the private key of the device. F_i represents the status information of the record (0 means public, 1 means secret), K_i represents the session key to be pushed this time, C_i is the ciphertext of K_i, C_i' is the preprocessed ciphertext of C_i. List2 is similar to list1, it is for jth IoT device and psk_j is the preprocessing key. The internal attacker is more powerful than the external attacker. If the internal attacker cannot attack successfully, then the external attacker also fails.

We consider the attacks from the internal attacker (adversary) as following two cases:

Case1: The adversary A does not have the private key usk of any legitimate IoT devices, but he owns the preprocessing key psk of an edge computing node for a legitimate device. So, he can get the preprocessing C'. His goal is to decrypt C' and get the session key K.

Case2: The adversary A owns the private key usk of a legitimate IoT device, but he has no legitimate status, so the edge computing node will not preprocess ciphertext for A. However, the adversary A goes for the plaintext K.

Init: The challenger C initializes the list1 and list2, and accepts the challenges from the adversary A.

Setup: The challenger C selects a safe parameter l and runs the system initialization algorithm $Setup(l)$ to generate system public parameter, then sends the parameter to adversary A.

*Phase*1: The adversary repeats the following inquiries. The adversary A chooses a legitimate device. The challenger C simulates the processes of subscription and push.

For case 1, assume an edge computing node can be monitored by the adversary. During the processes of simulating subscription and push, C records the necessary information into $List1=\{G_j, usk_{j,n}, 1, K_j, C_j, C_j'\}$ and $List2=\{G_j, psk_{j,n}, 0, K_j, C_j, C_j'\}$. Case2 is similar to case1.

Challenge : The adversary A chooses two different keys K_0 and K_1 have never been asked from the session key space, and submits them to the challenger. C chooses a random bit $b \in \{0,1\}$ by tossing a coin, and runs the encryption algorithm on K_b to generate ciphertext C_b, then returns the ciphertext to the adversary A.

*Phase*2 : Repeat *Phase*1 .

Guess : C publishes all the public items in list1 and list2 to the adversary, A outputs the guess result b' . If $b'=b$, the guess is correct and the adversary wins the game. Define the advantage of the adversary wins above game as

$$Adv_A = \Pr[b' = b] - 1/2 \quad (1)$$

If the advantage is ignored, we can say the encryption algorithm is secure under Chosen-Plaintext Attack (CPA).

5. Blockchain-based secure subscription and push service scheme

The scheme includes three entities: multiple Internet of Things devices, multiple edge computing nodes, and a service provider. A virtual component, public blockchain, is composed of the three entities. Edge computing nodes maintained the blockchain as miners using their partial computing power and storage, while IoT devices and service provider send transactions and deploy smart contract on the chain only. We design a smart contract called subscription service contract deployed by the service provider in the scheme. At the same time, our one-to-many encryption algorithm is embedded into the architecture to achieve the confidentiality of push messages and control the receivers' access authority.

There are four steps in our scheme, as described in the system architecture. The specific construction is as follows:

$Setup(k)$: System parameters are established by the service provider. Given a security parameter k , service provider chooses two groups G_1 and G_2 of prime order p , the generator of G_1 is g , selects a bilinear mapping $e: G_1 \times G_1 \to G_2$. The public parameters of the system are $Params = \{g, G_1, G_2, e\}$. U_i represents an IoT device is going to subscribe to a service, ECN_k represents the nearest edge computing node to the IoT device U_i . SP is the service provider. Suppose that the SP has published service names, their charges and other information on the blockchain for devices to select. Then the subscription service contract is deployed on the blockchain by SP in order to complete the automatic and fair trade.

Step P1. An IoT device subscribes to a service based on blockchain

U-Subscribe($Params$) : IoT device U_i selects a certain service S_j to subscribe after querying on the blockchain, calculates its identity information $id_i = H(addr_i)$ using collision resistant hash function $H(\cdot)$, for $addr_i$ is the address of U_i . Then it sends $\{id_i, S_j\}$ and the service prepayment to the contract. After a successful prepayment, U_i selects its private key $usk_{i,j} = x_{i,j} \in Z_p^*$, calculates its public key $R_{i,j} = g^{x_{i,j}}$, and sends $\{R_{i,j}, S_j, id_i\}$ to the nearest edge computing node ECN_k .

ECN-agentsub($R_{i,j}, id_i, S_j$) : ECN_k continues the subscription process as a proxy of device U_i . ECN_k verifies whether the IoT device has prepaid on the contract after receiving the subscription request from U_i . If the prepayment is successful, ECN_k selects a private key

$psk_{i,j} = k_{i,j} \in Z_p^*$ for U_i randomly according to the service S_j, and keep it secretly. The key will be utilized to pre-decrypt ciphertext for the IoT device later. Then ECN_k computes $E_{i,j} = R_i^{k_{i,j}}$ and sends $\{E_{i,j}, S_j, id_i\}$ to SP.

SP-Re$sponse(E_{i,j}, id_i, Params, \tau_j, y_j)$: SP processes subscription request and gives a response. SP computes $\omega_{i,j} = g^{\frac{1}{id_i + \tau_j}}$ and $T_{i,j} = g^{x_{i,j} k_{i,j} y_j - \tau_j} = E_{i,j}^{\ y_j} / g^{\tau_j}$ after receiving subscription message $\{E_{i,j}, S_j, id_i\}$ about U_i from agent ECN_k, $\tau_j \in Z_p^*$ is a key of service provider to generate subscription tag for service S_j to all the IoT devices, and is kept by the service provider secretly. $y_j \in Z_p^*$ is the transmission encryption key of service provider for service S_j, which is kept secretly by the service provider. Then, the service provider responds to ECN_k with $\{id_i, s_j, \omega_{i,j}, T_{i,j}\}$, and records the key information $\omega_{i,j}$ on the blockchain by a transaction as an evidence of U_i subscripting to S_j successfully. The form on chain is like $\{id_i, S_j, \omega_{i,j}\}$, and he builds a storage array for all subscription evidences, called SubscribeTag [$\omega_{i,j}$].

ECN-Re$sponse(\omega_{i,j}, T_{i,j}, psk_{i,j})$: ECN_k processes the response message about the subscription. After receiving the subscription response $\{id_i, S_j, \omega_{i,j}, T_{i,j}\}$ from SP, ECN_k adds it to the list created for service S_j, $\omega_{i,j}$ is the subscription tag of U_i for service S_j and $T_{i,j}$ is the information required to preprocess a ciphertext for U_i. The storage form is shown in **Fig. 2**. Finally, a message of successful subscription is sent to the IoT device U_i by ECN_k.

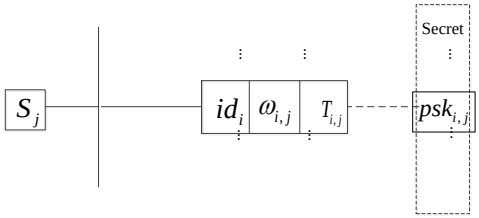

Fig. 2. The storage list of subscription tags by edge computing nodes

Step P2. The service provider pushes session key to legitimate IoT devices

SP-push-sessionkey$(K, y_j, Params)$: A session key (symmetric key) K will be pushed to all legitimate IoT devices after being encrypted by SP. Given a symmetric encryption key K, SP selects $t \in Z_p^*$ randomly, and computes ciphertext $C = \{c_1, c_2, c_3\}$, $c_1 = g^t$, $c_2 = g^{\tau_j t}$ and $c_3 = K \cdot e(g,g)^{y_j t}$. Then the service provider sends the push ciphertext C to ECN by broadcasting.

ECN-*CheckValidity-preprocess*$(C, \omega_{i,j}, psk_{i,j}, id_i, Params)$: After the ciphertext received by edge computing node ECN_k which is nearest to the IoT device U_i, it verifies whether the device U_i has subscribed to the service S_j. The specific process is as follows: extracts the id_i

and corresponding tag $\omega_{i,j}$ of U_i from the storage list, verifies whether the equation

$$e\left(c_2 c_1^{id_i}, \omega_{i,j}\right) = e\left(c_1, g\right) \tag{2}$$

holds or not. If yes, the node ECN_k sends $\left(S_j, \omega_{i,j}, id_i\right)$ to the Verify Subscribe algorithm to verify the legitimacy of the IoT device U_i, please see algorithm1 for details. ECN_k verifies whether $\omega_{i,j}$ exists on the blockchain and within the validity period. If the verification passes, the fixed fee would be deducted from the account of U_i and transfer to the account of SP through the smart contract automatically. Then ECN_k extracts $T_{i,j}$ and $psk_{i,j}$ corresponding to id_i in the list and computes $c_4 = \left(e\left(T_{i,j}, c_1\right) \cdot e\left(g, c_2\right)\right)^{\frac{1}{psk_{i,j}}}$ for U_i to reduce its computational burden, and sends the preprocessed ciphertext $C' = \{c_3, c_4\}$ to U_i. If one of verifications from (2) and smart contract fails, the subsequent operations are terminated.

U-*Decrypt*(C',usk) : After receiving the ciphertext C', U_i calculates the following equation to get the session key K,

$$K = \frac{c_3}{c_4^{\frac{1}{x_{i,j}}}} \tag{3}$$

The role of algorithm1is to verify the subscription tag and deducts fees from IoT device U_i automatically, called and executed by edge computing node ECN_k. The parameters need to be upload are subscription tag, service name, and the identity of IoT device.

Algorithm1: VerifySubscribe

Input : $id_i, S_j, \omega_{i,j}$

Output : bool

```
if msg.sender is not in ECN then
    throw;
end
if SubscribeTag[ω_{i,j}] is not in SubscribeTag then
    return false;
end
if SubscribeTag[ω_{i,j}].ID!= id_i then
    return false;
end
if Prepay[ S_j, id_i ].value < S_j 's price then
    return false;
end
transfer S_j 's price to SP;
Prepay[ S_j, id_i ].value = Prepay[ S_j, id_i ].value - S_j 's price;
Prepay[ S_j, id_i ].Time = Now;
return true;
```

Step P3. *SP* pushes encrypted service by broadcasting

The service provider pushes ciphertext of service message $c_5=\mathrm{Enc}_K(M)$ by broadcasting to IoT devices regularly. Legitimate IoT devices do symmetric decryption operation $M=Dec_K(c_5)$ and obtain service information.

So far, the subscription and push service are completed, and the revocation algorithm of the service is introduced below.

Step P4. blockchain-based service revocation.

The revocation is mainly achieved by deleting the subscription tag of a device. Without the subscription tag, the subscription verification cannot be passed, so the device cannot get the pre-decrypted ciphertext, and thus unable to obtain session key. In this way, it is convenient for the device to subscribe again.

Situation1: The subscription service expires, the subscription tag will not pass the verification conducted by smart contract, and the device would not receive the session key.

Situation2: The subscription service does not expire, but the IoT device applies for revocation. The process is as follows:

U-Revoke(*void*): The IoT device U_i selects the service S_j to be revoked, and sends a revocation request $\{id_i, S_j\}$ to smart contract and ECN_k.

ECN-agentRevoke(*void*) : ECN_k processes the revocation request. ECN_k sends the request and $\{id_i, S_j\}$ to SP.

SP-Revoke(*void*) : SP processes the revocation request. Service provider computes $C=\{c_1,c_2\}$ after receiving revocation message about the service S_j from ECN_k, and sends C to ECN_k, $t\in Z_p^*$ is random, and $c_1=g^t, c_2=g^{\tau_j t}$.

ECN-Re*sp*Revoke($C,\omega_{i,j},id_i,Params$): ECN_k processes the revocation response message. After receiving C from SP, ECN_k extracts $\omega_{i,j}$ and the corresponding id_i from the subscription tag storage list, and verifies whether the equation $e\left(c_2c_1^{id_i},\omega_{i,j}\right)=e(c_1,g)$ holds or not. If yes, ECN_k deletes $\omega_{i,j}, T_{i,j}$ and $psk_{i,j}$ from the subscription tag storage list. At last, the subscription tag $\omega_{i,j}$ is deleted by a transaction on the blockchain after smart contract confirms that the device has submitted a revocation request, and refund transaction is completed simultaneously. ECN_k notifies U_i that the service S_j has been revoked successfully.

6. Correctness and Security Analysis of the scheme

6.1 Correctness of our encryption algorithm

The correctness of the equation used for verifying the subscription tag:

For $\omega_{i,j}=g^{\frac{1}{id_i+\tau_j}}, c_2=g^{\tau_j t}, c_1=g^t$, then $e\left(c_2c_1^{id_i},\omega_{i,j}\right)=e\left(g^{\tau_j t}g^{id_i t},g^{\frac{1}{id_i+\tau_j}}\right)=e(g,g)^t$,

so, $e(c_1,g)=e\left(g^t,g\right)=e(g,g)^t=e\left(c_2c_1^{id_i},\omega_{i,j}\right)$.

The correctness of the equality used for decrypting the session key:

For $T_{i,j} = g^{x_{i,j}k_{i,j}y_j - \tau_j}$, $psk_{i,j} = k_{i,j} \in Z_p^*$, $c_3 = K \cdot e(g,g)^{y_j t}$, $c_4 = \left(e\left(T_{i,j}, c_1\right) \cdot e\left(g, c_2\right)\right)^{\frac{1}{psk_{i,j}}}$

$= \left(e(g,g)^{(x_{i,j}k_{i,j}y_j - \tau_j)t} \cdot e(g,g)^{\tau_j t}\right)^{\frac{1}{psk_{i,j}}} = e(g,g)^{x_{i,j}y_j t}$ and we can get

$$\frac{c_3}{c_4^{\frac{1}{x_{i,j}}}} = \frac{K \cdot e(g,g)^{y_j t}}{e(g,g)^{y_j t}} = K \tag{4}$$

6.2 Security analysis

6.2.1 The security analysis against malicious edge computing node

1) If a malicious edge computing node has verified the legitimacy of a subscription tag, but does not preprocess the ciphertext for the IoT device in order to save computing power. Then the IoT device can detect the behavior. As the verification is completed, the smart contract will be automatically triggered to transfer money from the account of the IoT device to the account of service provider. It can be found by the IoT device that the money was transferred while the preprocessed ciphertext is not received.

2) If a malicious edge computing node collude with an IoT device and sends preprocessed ciphertext to the device without verification on the smart contract so that the service fees of the device will not be deducted. Then the service provider can find it, because subscription tag is not verified on the smart contract, the money will not transfer to the service provider by the smart contract.

3) In order to get K, a malicious edge computing nodes must know $x_{i,j}$, and it is impossible for him, because it is a discrete logarithm problem on finite field, which is difficult [38]. And it is impossible for him to try to get the K by analyzing the ciphertext, because our algorithm satisfies the Chosen-Plaintext Attack, the detailed proof will be given later in part 6.2.4 in case1.

6.2.2 The security analysis against malicious IoT device

1) This situation is the same as the 3) in part 6.2.1.

2) In this situation, our algorithm satisfies Chosen-Plaintext Attack, the detailed proof will be given later in part 6.2.4 in case2.

6.2.3 The security analysis against malicious service provider

If the service provider does not distribute subscription tag for a certain IoT device, then the smart contract will not deduct the subscription fee of the device and transfer it to the service provider. At the same time, a certain fine will be deducted from the service provider's account, which is not cost-effective for the service provider.

6.2.4 Security proof of proposed encryption algorithm

Theorem1: Against the two adversary situations described in the security model, the algorithm is secure under the Chosen-Plaintext Attack.

Proof：In order to prove the security of the encryption algorithm, we define an interactive game and simulate the chosen-plaintext attack in two situations in the security model.

Init: The challenger C initializes the list1 and list2, and accepts the challenges from the adversary A.

Setup: The challenger C selects a safe parameter l and runs the system initialization algorithm $Setup(l)$ to generate system public parameters $params=\{g,G_1,G_2,e\}$, then sends the $params$ to adversary A.

*Phase*1: The adversary repeats the following inquiries. Adversary A selects a legitimate IoT device, the challenger C simulates the processes of subscription and push. According to the case1, the adversary can eavesdrop on edge computing nodes. Firstly, the adversary A selects a legitimate IoT device U_i randomly. The challenger C simulates the IoT device to subscribe to service S_n. C selects the decryption key $usk_{i,n}=x_{i,n}$ randomly and checks whether there is a record of public inquiries about the IoT device U_i and the decryption key. If the record has existed, it would remind the challenger that the inquiry to the IoT device cannot proceed, please select another device to inquire. Else, challenger C computes $R_{i,n}=g^{usk_{i,n}}$, runs the algorithm ECN-agentsub$(R_{i,n},id_i,S_n)$, chooses a private key $usk_{i,n}=k_{i,n}\in Z_p^*$ randomly, and computes $E_{i,n}=R_{i,n}^{k_{i,n}}$ to get $\{E_{i,n},S_n,id_i\}$. Then the challenger C simulates the service provider to run the algorithm SP-Re$sponse(E_{i,n},id_i,Params,\tau_n,y_n)$ to get $\{\omega_{i,n},T_{i,n}=g^{x_{i,n}k_{i,n}y_n-\tau_n}\}$. For the adversary A, he or she can know $\{R_{i,n},id_i,\omega_{i,n},psk_{i,n},T_{i,n}\}$. Secondly, the adversary A is going to inquire the process of pushing a session key. A sends a random key K_i chosen from the symmetric key space to C. After receiving the key, C checks if there is a public inquiry record of K_i. The inquiry would not be continued if it exists, and A must choose another key. Otherwise, C runs the algorithm SP-push-session$(K,y_n,Params)$ to get a ciphertext $C_i=\{c_{1,i},c_{2,i},c_{3,i}\}$ and sends it to A, for $c_{1,i}=g^{t_i}$, $c_{2,i}=g^{\tau_n t_i}$, $c_{3,i}=K_i\cdot e(g,g)^{y_n t_i}$. A can monitor the process of ciphertext C_i is preprocessed, and get the processed ciphertext C_i'. Finally, C makes two lists to record the necessary information during above processes, they are $List1=\{G_i,usk_{i,n},1,K_i,C_i,C_i'\}$ and $List2=\{G_i,psk_{i,n},0,K_i,C_i,C_i'\}$ separately.

According to the case2, the adversary A can impersonate a legitimate IoT device to inquire. Suppose he or she chooses a legitimate IoT device U_j. Then the challenger C simulates the IoT device U_j to subscribe to service S_n. The adversary A chooses a decryption key $usk_{j,n}$ for himself, and computes $R_{j,n}=g^{usk_{j,n}}$. C runs the algorithm ECN-agentsub$(R_{j,n},id_j,S_n)$, and selects a private key $psk_{j,n}=k_{j,n}\in Z_p^*$ to compute $E_{j,n}=R_{j,n}^{k_{j,n}}$ and get $\{E_{j,n},S_n,id_j\}$, then simulates service provider to run SP-Re$sponse(E_{j,n},id_j,Params,\tau_n,y_n)$ to get $\{\omega_{j,n},T_{j,n}=g^{x_{j,n}k_{j,n}y_n-\tau_n}\}$. Until now, A has got $\{R_{j,n},id_j,\omega_{j,n},usk_{j,n},T_{j,n}\}$. After those preparations, A starts the inquiry for the process of push. He selects a session key K_j from key space randomly and sends it to C. After receiving the key, C checks whether there is a public

record about the inquiry of K_j in list2. It indicates that the inquiry would not be continued if it exists, A must choose another session key. Otherwise, do the following procedure, C runs the algorithm SP-push-session($K, y_n, Params$) to get the ciphertext $C_i = \{c_{1,i}, c_{2,i}, c_{3,i}\}$. Then C performs the ciphertext preprocessing as an edge computing node, and sends the result C_j' to A. Finally, C creates two lists to record the necessary information during the entire process, $List1 = \{G_j, usk_{j,n}, 1, K_j, C_j, C_j'\}$ and $List2 = \{G_j, psk_{j,n}, 0, K_j, C_j, C_j'\}$ separately.

Challenge: The adversary A randomly chooses two keys K_0 and K_1 have not been asked before and submits them to the challenger C. C picks a bit $b \in \{0,1\}$ randomly by tossing a coin, and generates a ciphertext C_b by running SP-push-session($K_b, y_n, Params$), then returns the result to A.

*Phase*2: Repeat *Phase*1 as required and polynomial times at most.

Guess: C publishes all the open items in list1and list2 to the adversary A. A outputs his guess b'. If $b'=b$, then the guess is right, and the adversary A wins the game this time.

Suppose it has q_0 and q_1 times inquiries to case1 and case2 respectively, for $q_0, q_1 << p$. C gives all the open items in inquiry records to A, they are $List1 = \{G_i, usk_{i,n}, 0, K_i, C_i, C_i'\}$ and $List2 = \{G_j, psk_{j,n}, 0, K_j, C_j, C_j'\}$. And the adversary A gets the challenge ciphertext $C_b = \{c_{1,b}, c_{2,b}, c_{3,b}\}$, for $c_{1,b} = g^{t_b}$, $c_{2,b} = g^{\tau_n t_b}$, $c_{3,b} = K_b \cdot e(g,g)^{y_n t_b}$. Given the message K_0 and K_1, the probabilities of the adversary works out $c'_{3,b}$ and $c'_{3,\bar{b}}$ are all $1/2$, for $c'_{3,b} = e(g,g)^{y_n t_b}$, $c'_{3,\bar{b}} = D \cdot e(g,g)^{y_n t_b}$ and $\bar{b} \in [0,1]$, $\bar{b} = b+1$, $D = K_b / K_{\bar{b}}$ is a constant.

Given $c_{1,b} = g^{t_b}$, it is difficult to distinguish $c'_{3,b}$ from $c'_{3,\bar{b}}$ by the challenging ciphertext C_b for the adversary.

On the one hand, it is impossible to have a collision on the same IoT device and the same message about the public inquiries recorded in list1 andlist2, for $K_i \notin List2$, $K_j \notin List1$, $K_b \notin (List1 \vee List1)$. So, the adversary A cannot obtain the private key of a legitimate IoT device and private key of edge computing node nearest to the IoT device at the same time. On the other hand, the useful information to distinguish C_b from others in list1 and list2 published are $c_1 = \{c_{1,i}, c_{1,j}\}$, for $c_{1,i} = g^{t_i}$, $c_{1,j} = g^{t_j}$, $c_3 = \{c_{3,i}, c_{3,j}\}$, for $c_{3,i} = K_i \cdot e(g,g)^{y_n t_i}$, $c_{3,j} = K_j \cdot e(g,g)^{y_n t_j}$ and $c_4 = \{c_{4,i}, c_{4,j}\}$, for $c_{4,i} = e(g,g)^{usk_{i,n} y_n t_i}$, $c_{4,j} = e(g,g)^{usk_{j,n} y_n t_j}$. It is difficult to distinguish $g^{y_n t_i}|_{i \in List1}$, $g^{y_n t_j}|_{j \in List2}$ and $g^{y_n t_b}$ when given $c_1 = \{c_{1,i}, c_{1,j}\}|_{i \in List1, j \in List2}$ and $c_{1,b} = g^{t_b}$, which is as hard as distinguishing $e(g,g)^{y_n t_i}|_{i \in List1}$, $e(g,g)^{y_n t_j}|_{j \in List2}$, $c'_{3,b}$ and $c'_{3,\bar{b}}$. That is equal to the DDH problem.

So, it is not useful to know the open items in list1 and list2 to distinguish C_b for the adversary.

According to the above analysis, the probability of the adversary A wins the challenge is $1/2$, whose advantage is negligible. So, the scheme satisfies the security under the Chosen-Plaintext Attack.

7. Experimental evaluation

7.1 Experimental analysis of encryption and decryption algorithms

We compare the algorithm of this paper with the schemes [20] and [27] in reference, and perform simulation evaluation. The hardware environment as: 3.20GHZ Inter(R)Core(TM) i7-8700 CPU@3.20GHz and RAM16.0GB. The software environment is: operating system is Windows10 and VC++ 6.0PBClibrary.

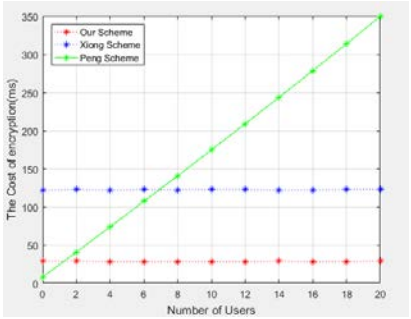

Fig. 3. Comparison of encryption efficiency

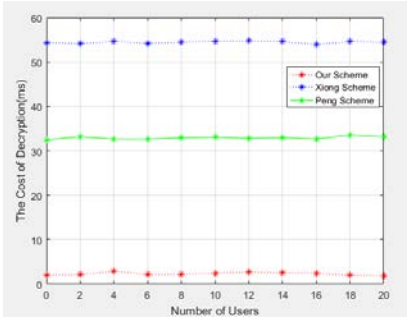

Fig. 4. Comparison of decryption efficiency

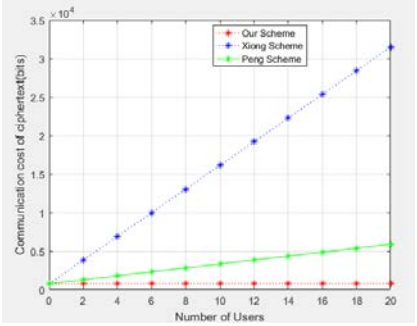

Fig. 5. Comparison of ciphertext length (communication efficiency)

Experiment results shows that the time costs of encryption and decryption in our algorithm do not change with the increase of the number of IoT devices, which can see from **Fig. 3** and **Fig. 4**. Compared with the schemes in [20] and [27], it maintains a higher computational efficiency. Especially, the decryption computation $2P + PM + E$ is borne by edge computing nodes. When performing registration, IoT device only executes hash and exponent operation that require about 0.504ms, and only exponent E and inversion operations are executed when decrypting that require about 0.497ms. They are all the operations suitable for IoT devices. Where P represents a pair operation, PM is once scalar multiplication and E represents an exponent operation. **Fig. 5** shows that the length of the ciphertext in our article will not increase with the quantity of IoT devices growing. Compared with the solutions of [20] and [27], the communication efficiency is higher, and is more suitable for multiple IoT device groups.

7.2 Efficiency comparison of overall efficiency

We chose reference [27] as the centralized scheme which is closest to ours in order to compare the overall efficiency. Registration payment and the cancellation of subscription are only executed once, so we compared the efficiency except for these two parts. And the comparison

of overall efficiency between the two schemes completed in the simulation experiments and the hardware environment is: 3.20GHZ Inter(R)Core(TM) i5-3470 CPU@3.20GHz and RAM4.0GB, VC++ 6.0PBC library was called, and we chose the identical curve $y^2 = x^3 + 7$ for testing in the two schemes.

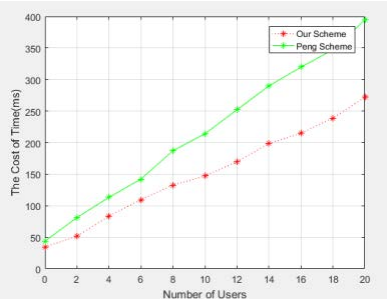

Fig. 6. Comparison of overall efficiency

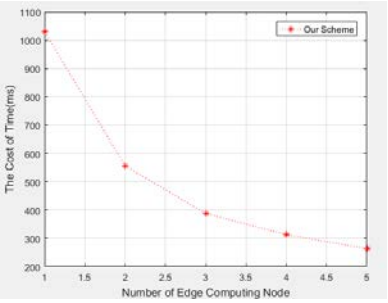

Fig. 7. Efficienciesof different number of edge nodes

Fig. 6 shows the result in the case of five edge computing nodes. Compared with Peng's scheme, when the number of users is growing from 0 to 20, our proposed scheme takes less time than Peng's, and the time increases at a slow rate. So, the efficiency of our solution is better than Peng's. The **Fig. 7** shows a simulation result of entire system in the case of the number of users is fixed at 20, the users distributed around edge computing nodes uniformly and the number of edge computing nodes is from 1 to 5.

7.3 Gas cost

We also give the execution efficiency of the smart contract, the gas cost of every step is presented in **Fig. 8**. The result is from the Ethereum testnet （http://remix.ethereum.org ）, the unit is gas, and 1gas=10^{-8}Ether.

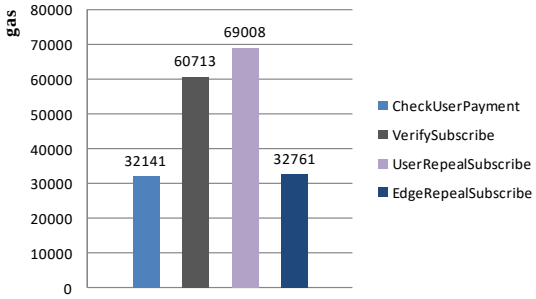

Fig. 8. The gas cost of smart contract

Experimental result shows the cost of our smart contract is reasonable.

8. Conclusion

It is an important research content in the Internet of Things environment that the IoT devices receive interested information timely. Based on blockchain technology and smart contract, a decentralized subscription-push service scheme is implemented using the idea of broadcast encryption. Safe and timely information push has realized for IoT devices, and a detailed

description of the entire push process and safety analysis are included in this article. Because of the low computing power of IoT devices, in our encryption algorithm, we introduced edge computing to share part of the decryption operation, made the calculation operations are suitable for the IoT devices. The rigorous proof guarantees the security of the encryption algorithm. Finally, the encryption algorithm is evaluated through experiments, the results have proven that our algorithm is more efficient and we performed a simulation for the entire scheme, the result shows that our program is better than the Peng's with the number of the IoT devices is growing.

References

[1] S. Li, L. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243-259, Apr. 2015. Article(CrossRef Link)

[2] Z. Pan, X. Liang, Y. Zhou, Y. Ge and G. Zhao, "Intelligent Push Notification for Converged Mobile Computing and Internet of Things," in *Proc. of IEEE International Conference on Web Services*, New York, USA, pp. 655-662, 2015. Article (CrossRef Link)

[3] P. T. Eugster, P. Felber, R. Guerraoui, and A. M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys*, vol. 35, no. 2, pp. 114-131, Jun. 2003. Article (CrossRef Link)

[4] R. Vasconcelos, A. Carvalho and A. Carrapatoso, "One-to-many reliable data distribution using multiple multicast groups," in *Proc. of the 24th Annual Conference of the IEEE Industrial Electronics Society*, Aachen, Germany, pp.173 -175, 1998. Article (CrossRef Link)

[5] A. Fiat and M. Naor, "Broadcast encryption," in *Proc. of Advances in Cryptology-CRYPTO' 93*, Santa Barbara, California, USA, pp. 480-491, 1993. Article (CrossRef Link)

[6] M. Bellare, A. Boldyreva and S. Micali, "Public-Key Encryption in a Multi-user Setting: Security Proofs and Improvements," in *Proc. of EUROCRYPT 2000*, Bruges, Belgium, pp. 259-274, 2000. Article (CrossRef Link)

[7] K. Paridel, Y. Vanrompay, and Y. Berbers, "Fadip: Lightweight Publish/Subscribe for Mobile Ad Hoc Networks," in *Proc. of On the Move to Meaningful Internet Systems 2010*, Crete, Greece, pp. 798-810, 2010. Article (CrossRef Link)

[8] W. Su, W. Chen and C. Chen, "An Extensible and Transparent Thing-to-Thing Security Enhancement for MQTT Protocol in IoT Environment," in *Proc. of 2019 Global Internet of Things Summit*, Bruges, Belgium, pp. 1-4, 2019. Article (CrossRef Link)

[9] T. D. Nguyen, E. N. Huh, and M. Jo., "Decentralized and Revised Content-Centric Networking-Based Service Deployment and Discovery Platform in Mobile Edge Computing for IoT Devices," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4162-4175, Jun. 2019. Article(CrossRef Link)

[10] L. Zhu, N. M. R. Lwamo, K. Sharif, C. Xu, D. Du, M. Guizani and F. Li, "T-CAM: Time-based content access control mechanism for ICN subscription systems," *Future generation computer systems*, vol. 106, pp. 607–621, May. 2020. Article (CrossRef Link)

[11] M. Wazrd, A. K. Das, S. Shetty and M. Jo, "A Tutorial and Future Research for Building a Blockchain-Based Secure Communication Scheme for Internet of Intelligent Things," *IEEE Access*, vol. 8, pp. 88700-88716, May. 2020. Article(CrossRefLink)

[12] D. Boneh, X. Boyen and E. J. Goh, "Hierarchical Identity Based Encryption with Constant Size Ciphertext," in *Proc. of EUROCRYPT 2005*, Aarhus, Denmark, pp. 440-456, 2005. Article (CrossRef Link)

[13] D. Boneh, C. Gentry and B. Waters, "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys," in *Proc. of CRYPTO2005*, Santa Barbara, California, USA, pp. 258-275, 2005. Article (CrossRef Link)

[14] C. Delerablée, "Identity-based broadcast encryption with constant size ciphertexts and private keys," in *Proc. of ASIACRYPT2007*, Kuching, Malaysia, pp. 200-215, 2007. Article (CrossRef Link)

[15] Y. Liu and W. G. Tzeng, "Public Key Broadcast Encryption with Low Number of Keys and Constant Decryption Time," in *Proc. of Public Key Cryptography 2008*, Barcelona, Spain, pp. 380-396, 2008. Article (CrossRef Link)
[16] A. Lewko, A. Sahai and B. Waters, "Revocation Systems with Very Small Private Keys," in *Proc. Of 2010 IEEE Symposium on Security and Privacy*, Oakland, CA, USA, pp. 273-285, 2010. Article (CrossRef Link)
[17] X. Du, Y. Wang, J. Ge, and Y. Wang, "An ID-based broadcast encryption scheme for key distribution," *IEEE Transaction on broadcasting*, vol. 51, no. 2, pp. 264-266, Jun. 2005. Article (CrossRef Link)
[18] S. Canard, D. H. Phan, and V. C. Trinh, "Attribute-based broadcast encryption scheme for lightweight devices," *IET Information Security*, vol. 12, no. 1, pp. 52-59, Jan. 2018. Article (CrossRef Link)
[19] T. V. X. Phuong, G. Yang, W. Susilo and X. Chen, "Attribute Based Broadcast Encryption with Short Ciphertext and Decryption Key," in *Proc. of Computer Security-ESORICS 2015*, Vienna, Austria, pp. 252-269, 2015. Ariticle (CrossRef Link)
[20] H. Xiong, H. Zhang and J. Sun, "Attribute-Based Privacy-Preserving Data Sharing for Dynamic Groups in Cloud Computing," *IEEE Systems Journal*, vol. 13, no. 3, pp. 2739-2750, Sept. 2019. Article (CrossRef Link)
[21] A. Lewko and B. Waters, "Why Proving HIBE Systems Secure Is Difficult," in *Proc. of EUROCRYPT 2014*, Copenhagen, Denmark, pp. 58-76, 2014. Article (CrossRef Link)
[22] Y. M. Tseng, T. T. Tsai and T. Y. Wu, "Efficient Revocable Multi-Receiver ID-Based Encryption," *Information technology and control*, vol. 42, no. 2, pp. 159-169, May. 2013. Article (CrossRef Link)
[23] D. He, H. Wang, L. Wang, J. Shen and X. Yang, "Efficient Certificateless Anonymous Multi-Receiver Encryption Scheme for Mobile Devices," *Soft Computing*, vol. 21, no. 22, pp. 6801-6810, Nov. 2017. Article (CrossRef Link)
[24] Y. H. Hung, S. Huang, Y. M. Tseng and T. T. Tsai, "Efficient Anonymous Multi-Receiver Certificateless Encryption," *IEEE Systems Journal*, vol. 11, no. 4, pp. 2602-2613, Dec. 2017. Article (CrossRef Link)
[25] L. Deng, "Anonymous Certificateless Multi-Receiver Encryption Scheme for Smart Community Management Systems," *Soft Computing*, vol. 24, no.1, pp. 281-292, 2020. Article (CrossRef Link)
[26] L. Pang, M. Kou, M. Wei and H. Li, "Efficient Anonymous Certificateless Multi-Receiver Signcryption Scheme without Bilinear Pairings," *IEEE Access*, vol. 6, pp. 78123-78135, Dec. 2018. Article (CrossRef Link)
[27] C. Peng, J. Chen, M. S. Obaidat, P. Vijayakumar and D. He, "Efficient and Provably Secure Multi-Receiver Signcryption Scheme for Multicast Communication in Edge Computing," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6056-6068, Jul. 2020. Article (CrossRef Link)
[28] D. Boneh, "The decision Diffie-Hellman problem," in *Proc. of International Algorithmic Number Theory Symposium*, Portland, USA, pp. 48-63, 1998. Article (CrossRef Link)
[29] S. Lee, J. Lee, S. Hong and J. H. Kim, "Lightweight End-to-End Blockchain for IoT Applications," *KSII Transaction on Internet and Information Systems*, vol. 14, no. 8, pp. 3224-3242, Aug. 2020. Article (CrossRef Link)
[30] S. Underwood, "Blockchain beyond Bitcoin," *Communication of the ACM*, vol. 59, no.11, pp. 15-17, Nov. 2016. Article (CrossRef Link)
[31] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu and J. Kishigami, "Blockchain contract: Securing a blockchain applied to smart contracts," in *Proc. of IEEE International Conference on Consumer Electronics*, Las Vegas, NV, USA, pp. 467-468, 2016. Article (CrossRef Link)
[32] L. Luu, D. H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proc. of 2016 ACM Conference on Computer and Communications security*, Vienna, AT, pp. 254-269, 2016. Article (CrossRef Link)

[33] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637-646, Oct. 2016. Article (CrossRef Link)
[34] K. Xue, P. He, X. Zhang, Q. Xia, D. S. L. Dei, H. Yue and F. Wu, "A Secure, Efficient, and Accountable Edge-Based Access Control Framework for Information Centric Networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp.1220-1233, Jun. 2019. Article (CrossRefLink)
[35] S. C. Inés, R. S. Alonso, M. C. Juan, R. G. Sara and C. V. Roberto, "A review of edge computing reference architectures and a new global edge proposal," *Future Generation Computer Systems*, vol. 99, pp. 278-294, Oct. 2019. Article (CrossRef Link)
[36] A. Bogdanov, M. Knezevic, G. Leander, D. Toz, K. Varici and I. Verbauwhede, "SPONGENT: The Design Space of Lightweight Cryptographic Hashing," *IEEE Transaction on Computers*, vol. 62, No.10, pp. 2041-2052, Oct. 2013. Article (CrossRef Link)
[37] B. Ying, and A. Nayak, "Anonymous and Lightweight Authentication for Secure Vehicular Networks," *IEEE Transactions on Vehicular Technology*, vol. 66, No. 12, pp. 10626-10636, Dec. 2017. Article (CrossRef Link)
[38] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469-472, Jul. 1985. Article (CrossRef Link)

Yin-juan Deng received her M.S degree in applied mathematics in 2013 from Xi'an University of Technology, Xi'an, China. Currently, she is pursuing the Ph.D degree in Xi'an University of Technology. Her current research interests are blockchain technology and privacy problem in Internet of Thing.

Shang-ping Wang received his B.S. degree in mathematics in 1982 from Xi'an University of Technology, Xi'an, China. He received his M.S. degree in applied mathematics in 1989 from Xi'an Jiaotong University, Xi'an, China, and earned his Ph.D degree in cryptography in 2003 from Xidian University, Xi'an, China. Currently, he is a professor in Xi'an University of Technology. His current research interests are blockchain technology and the safety problem in Internet of Thing.

Qian Zhang received her M.S. degree in 2019 from Xi'an University of Technology, Xi'an, China, and she is currently working toward the Ph.D degree from Xi'an University of Technology. Her current research interests include information security, cryptography and blockachin.

Duo Zhang received her M.S. degree in 2017 from Xi'an University of Technology, Xi'an, China, and currently, she is a Ph.D candidate in Xi'an University of Technology. Her current research interests include information security, modern cryptography and blockachin technology.