Novel lightweight and fine-grained fast access control using RNS properties in fog computing

Mohammad Ali Alizadeh[1], Somayyeh Jafarali Jassbi[1]*, Ahmad Khademzadeh[2] and Majid Haghparast[3]

[1.] Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

[2.] Iran Telecommunication Research Center, Tehran, Iran

[3.] Faculty of Information Technology, University of Jyväskylä, FI-40014, Jyväskylä, Finland

* Corresponding author, E-Mail addresses: s.jassbi@srbiau.ac.ir

Abstract

Fog computing provides a suitable development for real-time processing in the Internet of Things (IoT). Attribute-based encryption (ABE) is the main method to control data access in fog computing. A residue number system (RNS) can speed up multiplication and exponential operations by converting very large integers to small integers. This paper proposes a fine-grained lightweight access control scheme using ABE modified with RNS properties (RNS-ABE) with fog computing. Decryption is implemented with the Chinese remainder theorem (CRT), and a new access structure based on the CRT secret sharing scheme is also introduced. Security of the proposed scheme proved based on RNS properties and the complicated problem of factoring a very large integer into its large prime factors, like RSA encryption. The time cost comparison shows that the total encryption and decryption time of our scheme is more efficient than the lightweight schemes with the underlying operation of bilinear pairing.

Keywords: Attribute-based encryption, residue number system, chinese remainder theorem, fog computing, data access control, internet of things

1 Introduction

Today, the Internet of Things (IoT) plays a prominent role in the world of communication and information, and all equipment and machines in various fields, such as the environment, agriculture, home, transportation, and health, can become smart devices. To implement such a digital ecosystem, we need high security against various attacks, privacy protection, authentication, and access control. In this regard, it is necessary to make the required changes in the architecture of the IoT applications to achieve end-to-end secure environments in IoT [1]. Due to the connection of many devices with different applications, classical security methods such as public key cryptography are challenging to implement due to energy, memory, and processing limitations [2]. Fog computing paradigms are becoming a popular tool for optimal resource utilization by IoT devices and achieving fast processing in cloud-based IoT ecosystems. Fog computing overcomes the limitations of the cloud system by improving the robustness, efficiency, and performance of the cloud infrastructure [3]. Some applications depend on fast calculations, pre-processing, and filtering and must have high security and protect users' privacy. Fog computing helps cloud computing and IoT devices in this situation [4]. Access control (AC) is one of the preventive data protection requirements against unauthorized access. Choosing an appropriate, operational, and fine-grained AC scheme in an IoT ecosystem can reduce the time cost of processing various applications. One of the most essential tasks of fog computing in an AC scheme is the possibility of time-consuming outsourcing

calculations such as exponentiation and bilinear pairing belonging to devices with limited processing resources [5]. Attribute-based access control (ABAC) is functional and fine-grained and can interact well with fog computing. A data owner can specify an access policy for data users based on different attribute sets in the fog layer. Any user, resource, or environmental condition whose attribute set satisfies the access policy can access the fog server. Attributes can be a username, job, time, or last accessed. In this model, fog computing interacts with four components: IoT devices or data users, data owners, cloud infrastructure, and authorities [6]. ABAC is based on attribute-based encryption (ABE). One of the ABE types that are very practical and progressive is ciphertext-policy ABE (CP-ABE). CP-ABE provides better plasticity to data owners than other types. Time costs in the encryption and decryption operations of CP-ABE schemes have a linear relationship with the number of attributes. However, due to performing multi-exponential operations and bilinear pairing that has a heavy computational overhead, data owners and data users cannot perform calculations on their own in an adequate time with limited processing resources. As a result, they will have a mandatory dependence on fog computations. Therefore, most existing ABE schemes are unsuitable for data confidentiality and fine-grained access control [7]. Recently, residual number system (RNS) has been used extensively in public key cryptography, such as RSA and ECC, with the key length between 160 bits and 4096 bits to reduce computational overhead and increase efficiency. RNS uses small word-length processors in applications where multiple multiplication and addition operations must be performed efficiently on very large integers [8].

This feature can be used in ABE to simplify calculations and provide the RNS-ABE scheme. Based on this, the following innovations are presented in this article:

(1) Exponential and bilinear pairing calculations have not been used, and instead, calculations based on RNS, such as residue arithmetic and binary to RNS conversion, have been used.
(2) The length of the keys depends on the length of the plaintext, and therefore, the time cost of the RNS-ABE scheme can be reduced for the plaintext with a shorter length.
(3) It is possible to parallelize the encryption computations outsourced to the fog server and the computations by the data owner, and thus, it can be accelerated.
(4) A new agile access structure with independent sections has been introduced.

This paper is organized as follows. We review related work in Section 2. The underlying operations are introduced in section 3. In section 4, the security model, system architecture, and its entities are defined. Section 5 introduces the construction of the RNS-ABE scheme in detail. In section 6, the security of the scheme and its resistance to various attacks are analyzed. In section 7, the efficiency and time cost of the proposed scheme are calculated and compared with three other lightweight schemes. Finally, section 8 concludes.

2 Related work

Bettencourt et al. presented for the first time an operational model of CP-ABE based on bilinear mapping using pairing-based cryptography. The most important advantage of this one-to-many encryption scheme is that it does not require utterly reliable cloud computing. However, the main problem of this design is the lack of simplicity of user revocation or attribute revocation calculations. Also, this scheme is susceptible to collusion attacks [9]. In the paper [10], the authors introduce a scheme that can make CP-ABE efficient for IoT devices in terms of energy cost with an innovative precomputation technique. In fact, in this technique, many complex and expensive calculations have already been calculated and stored in the devices. Also, this paper has shown the achievements of

saving energy in terms of computational costs. He et al. in [11] have proposed a lightweight AC scheme for WSN-integrated cloud computing (WCC) called FLAC, which includes an encryption algorithm based on CP-ABE and a signature algorithm based on attribute-based signature (ABS) for data integrity. In this scheme, the calculations have been outsourced to the cloud infrastructure, which has problems for real-time calculations due to the lack of use of fog computing. In [12], an efficient AC scheme with the capability of outsourcing and attribute updating is proposed by applying fog computing. In article [13], which is the basis of article [14], the authors have proposed a multilayer architecture for the AA entities based on hierarchical attribute-based encryption (HABE). In addition, by using ABS, the authorized change of data is guaranteed. However, fog computing is not used in this article. Then, the authors in the article [14] present a secure and lightweight data outsourcing scheme and move most of the high-cost operations from IoT devices to fog nodes. It also provides the ability to update the ciphertext on the cloud infrastructure using an ABS. In this work [15], a safe and efficient e-learning scheme based on fog is introduced and subsequently integrates fog calculations into the e-learning system to reduce the latency of the e-learning services provided. Accordingly, it designs a low-cost ABE-IBBE hybrid design with support for connected devices. This paper [16] first introduces an attribute-based encryption model called matchmaking (MABE) with a collision-resistant hash function. It then uses MABE to create a data-sharing access control system in cloud computing. CP-ABE rarely focuses on user revocation and attribute revocation in fog computing and still imposes high computational overhead and storage on IoT devices with limited resources. So [17] offers an efficient encryption outsourcing scheme based on user revocation and attribute revocation for IoT using fog calculations. In the paper [18], Miao et al. presented a lightweight fine-grained ciphertext search system (LFGS) with fog computing architecture based on CP-ABE and searchable encryption (SE). By transferring calculations with high overhead to the fog node, this scheme can search and control access to the ciphertext with the help of keywords with proper efficiency.

Zhang et al. [19] present an AC scheme using hidden resources in cloud-fog computing that can provide data privacy, manageable outsourcing capabilities, and fine-grained access control. Internet of Vehicles (IoV) features automated driving and 5G communications, attracting unprecedented attention in academia and industry. Decryption with external sources in these designs still adopts a low-speed serial computation mode, creating a poor user experience, especially for users of the fog-intelligent IoV terminal. The parallel computation mode of outsourced decryption has been investigated, and its feasibility has been proved. As a result, Feng et al. [20] proposed a parallel outsourced decryption model for edge intelligent IoV ABEM-POD. The paper [21] proposed a combination of lightweight symmetric and asymmetric encryption algorithms based on proxy re-encryption (PRE) to increase data security with the help of fog computing in the IoT ecosystem. It has been tried to introduce a lightweight proxy re-encryption scheme with a lower computational cost. Electronic medical record (EMR) systems increase the efficiency of medical services, improve the performance of human resources (HR) and make the prescription of the drug more accurate. Due to the sensitivity of medical records, security issues in EMR subscription systems are of paramount importance. The scheme [22] proposed a lightweight access control for EMR sharing in cloud computing with fog computing cooperation. The scheme [23] introduces a dynamic update policy and specifies revocation processes with PRE. It also adopts a chaotic map to generate a one-time key for encryption. Saidi et al. in [24] presented the SHARE-ABE scheme based on CP-ABE, which provides a new collaborative approach to privacy protection. They have used

fog computing to outsource part of the overhead decryption operation. In fact, this scheme introduces a new model of a collaboration attribute that allows users within a group to combine their attributes and maintain privacy in the access policy by using false attributes. In the paper [25], the authors propose an efficient and secure multi-level scheme, MLS-ABAC, based on NIST's ABAC model, which improves privacy with verification operations.

3 Preliminaries

3.1 Residue number system (RNS)

As in [26], in RNS, first, the huge integer is converted into the set of remainders, which are small integers, based on forward conversion (binary to RNS), and after calculations on them, it is converted into another large integer by reverse conversion (RNS to Binary).

3.1.1 Forward conversion

(1) We choose a set of relatively small integers that are prime to each other and call it the moduli set $\{m_1, m_2, \cdots, m_n\}$.

(2) We take the remainder (residue) of an arbitrarily large integer X concerning the modules belonging to the moduli set with n length.

$$\{x_i \mid x_i = X \bmod m_i \ \ for\ 1 \le i \le n\} \tag{1}$$

3.1.2 Reverse conversion

Chinese Remainder Theorem (CRT) and Mixed Radix Conversion (MRC) are two main techniques for reverse conversion. Recently, new techniques based on these two methods, such as New CRT-I, New CRT-II, and Mixed-Radix CRT, have also been presented. MRC has a serial structure, and CRT has a parallel structure, and for its proper performance, our choice is CRT as

$$X = \sum_{i=1}^{n} x_i M_i \left(\frac{1}{M_i}\right)_{m_i} \bmod M \tag{2}$$

Where $M_i = M/m_i$ for i = 1, 2, …, n and $M = \prod_{i=1}^{n} M_i$. M is known as the dynamic range. $\left(\frac{1}{M_i}\right)_{m_i}$ is known as the multiplicative inverse of M_i. In other words,

$$\left(M_i \left(\frac{1}{M_i}\right)\right)_{m_i} = 1 \tag{3}$$

3.2 Access structure

The access structure (access policy) related to the RNS-ABE scheme is introduced for the first time. Considering that the proposed scheme is not based on the binary number system and bilinear pairing and uses RNS capabilities, it is necessary to design its special access structure. Obviously, an access structure specific to the binary number system cannot be implemented in another number system, such as RNS. According to state-of-the-art knowledge, such a structure has not been introduced so far. We call it the RNS access structure, and we show it in Fig. 1. Each attribute used in this structure, which has a variable length, must be converted into a 512-bit numeric string through the SHA3-512 hash function, which we will call the attribute module. During encryption, we calculate the remainder of a large integer that is the plaintext relative to the attribute modules. Ciphertext is also the set of residues relative to attribute modules. Because it is necessary for the modules to be prime in RNS, all attribute modules must be converted to an equivalent prime number. Therefore, we select the smallest prime number larger than the value of each attribute module and

call it the prime attribute module. In this scheme, which aims to implement fine-grained access

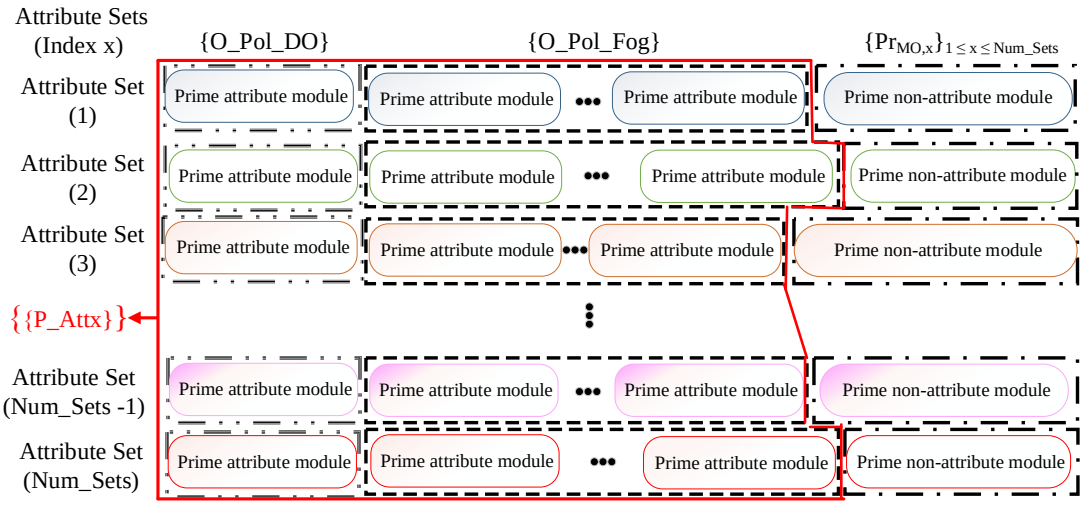

Fig. 1 RNS access structure

control using attribute-based cryptography, we consider attribute, attribute module, and prime attribute module to be equivalent for simplicity.

This structure is a two-dimensional array in which a valid set of prime attribute modules is loaded in each row, and in each column except the last column, a prime attribute module that is a member of that set can be loaded. The number of rows is equal to Num_sets, and the number of columns in each row can be different. If the attributes belonging to a data user exactly match the prime attribute modules of each line of the RNS access structure, it can satisfy the access structure, and we will see in the future it can decrypt the ciphertext. Each line of the access structure, or in other words, each attribute set that makes up the access structure, consists of three module models as follows:

- The first model has a fixed length of 512 bits and is converted into prime attribute modules by the data owner. These modules are all members of the *{O_Pol_DO}* set. This set must remain confidential with the data owner and will not be sent to the data user, fog, and cloud servers. This improves security, and if the adversary penetrates them, they are denied access to the entire access structure.
- The second model has a fixed length of 512 bits and is defined by the data owner. However, to produce the equivalent prime attribute modules, they are outsourced to the fog node to reduce the computational overhead. All these modules are members of *{O_Pol_Fog}* set. After both sets of *{O_Pol_DO}* and *{O_Pol_Fog}* are completed, the data owner designs part of the access structure and places them in the associated rows. We call it partial access structure *{{P_Attx}}*. Then it sends *{{P_Attx}}* to the attribute authority to receive the third model of required modules.
- The third model is neither an attribute module nor has a fixed length of 512 bits. The length of these modules should be proportional to the length of the plaintext and the length of each row of *{{P_Attx}}*. Therefore, random numbers with the desired length are generated, and subsequently, the attribute authority calculates the smallest prime number larger than them. We call them prime non-attribute modules $\{Pr_{MO,x}\}$, where *MO* is the public key of the data

owner. In the end, the attribute authority loads them at each end of each row of the access structure and sends them to the data owner. These modules are required to validate the reverse conversion CRT in our scheme and also ensure that no access structures are generated without the attribute authority supervision. In previous schemes based on CP-ABE, the attribute authority was not involved in the formation of the access structure, and the data owner produced this structure alone.

The RNS access structure has advantages and differences from the two famous access tree structures [14] and LSSS [27, 28], which we discuss below:

- Our proposed structure is based on RNS capabilities and consists of independent parts. Each of them is a set of attributes that can satisfy it. Each attribute set is composed of a number of attributes that play the role of modules in RNS. Therefore, we can perform part of the encryption and decryption calculations that require the use of the proposed access structure in parallel and reduce the time cost. However, the access tree and LSSS are compatible with the binary number system. This structure has been specially designed for RNS for the first time.
- The RNS access structure is not sent to the fog, cloud server, and data users and remains confidential with the data owner. Only a part of the calculations of the prime attribute modules is outsourced to the fog server, which cannot endanger the confidentiality and privacy of data users due to the lack of knowledge of their exact location in the proposed access structure.
- There is no need to perform calculations with a relatively high time cost to discover an attribute set in the decryption phase because each set of attributes is placed in it independently. However, in the access tree, during a recursive operation, non-leaf nodes consisting of threshold gates are traversed to discover an attribute set whether it can satisfy the access tree or not. In LSSS, the constants $\{\omega_i \in Z_p\}_{i\in I_s}$ that lead to the production of *(1 ,0, ..., 0)* should be discovered to confirm the validity of an attribute set.
- In LSSS, each row of the share-generating matrix is equivalent to an attribute, not a set of attributes. All columns of its matrix must be valued. In the RNS access structure, the total length of the prime attribute modules in all rows must be equal. This is required for a valid and unique CRT reverse conversion.
- The RNS access structure can be of variable size based on the plaintext length. However, the size and complexity of the access tree and LSSS have no dependence on the size of the plaintext. Our proposed structure is effective for devices with limited processing and storage resources that generate small volume data. However, it may not be efficient for plaintexts with a large volume because it is necessary to generate very large random prime numbers with a certain length. Of course, it is possible to reduce the overhead of calculating them with solutions such as pre-computing very large prime numbers with specific lengths instead of generating random prime numbers.

4 System architecture and security model

4.1 System architecture

The proposed architecture is somewhat similar to CP-ABE because the plaintext encryption is based on the access policy, and the private key decryption is based on the data user attributes. However, its main difference with CP-ABE is that RNS properties are used for the first time instead of bilinear mapping. The new model is called RNS-ABE. It has six independent entities, as shown in Fig. 2: Attribute authority (AA), Cloud server (CS), Communication fog server (Prime Fog), CRT fog server (CRT Fog), Data owner (DO), and Data user (DU). The task of each entity is as follows:

AA: This entity is fully trusted and validates, specifies, and records DO and DU attribute sets. It also generates the public key belonging to DO, the private key belonging to DU, and

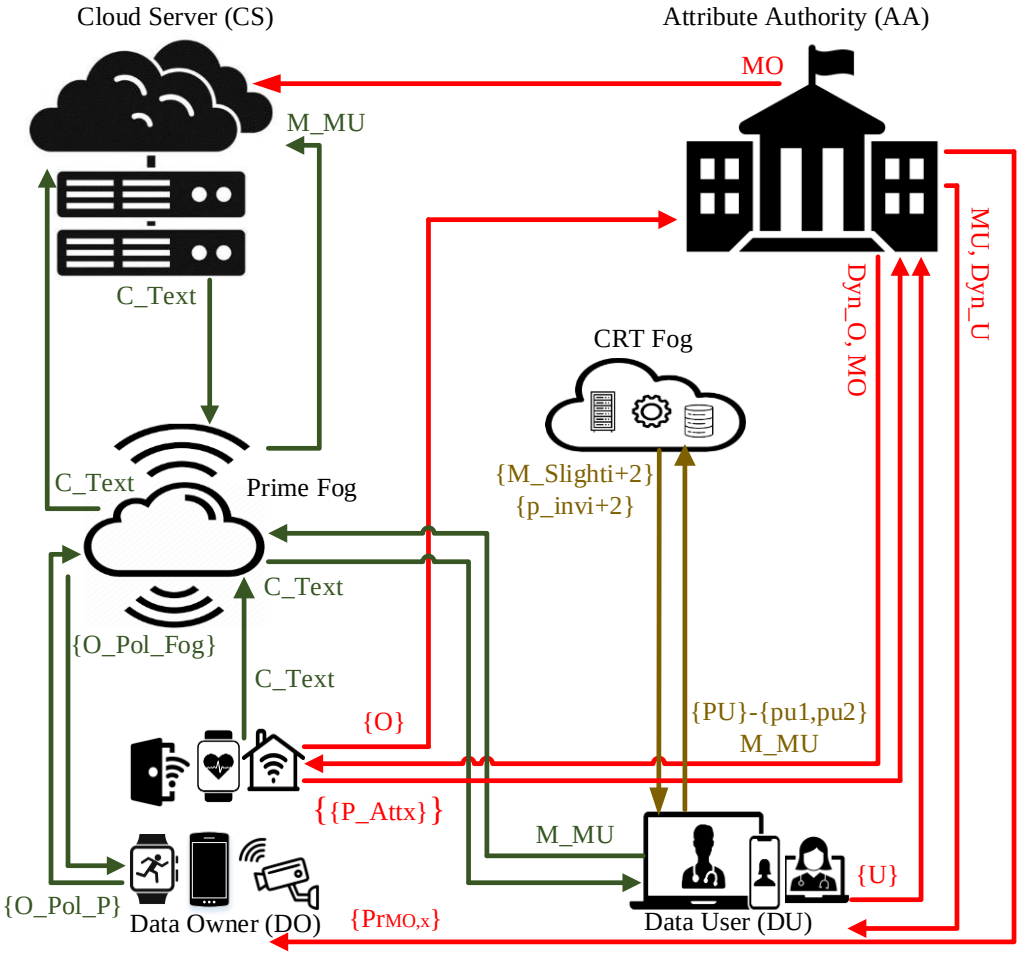

Fig. 2 System architecture

random numbers required by the access policy and sends them to the respective recipient through a secure channel.

CS: Firstly, it is responsible for storing the ciphertext sent from valid DOs, and secondly, it is responsible for sharing the ciphertext for valid DUs. It receives the ciphertext from the valid data owners through the Prime fog and sends the ciphertext to the Prime fog after satisfying the access policy defined by the DO for authorized DUs.

Prime fog: It helps DOs and DUs to send/receive ciphertext to/from the CS for sharing with less energy consumption and faster speed. It also helps to calculate the equivalent primes of the attributes used in the access policy during encryption by the DO.

CRT fog: Unlike Prime fog, it has no connection with CS and DOs and only communicates with

DUs. As a result, its security has increased due to the reduced access level. It is responsible for calculating the inverse of multiplication when converting data from RNS to a binary number system.
DO: First, it must send its attribute set to AA to receive its valid public key. In addition, it converts the data generated by sensors and devices into plaintext of the specified length. Then, encrypt it to ciphertext based on the access policy defined with the help of Prime fog and AA. The ciphertext, based on RNS, is displayed. Then, it sends it to Prime fog for the access of authorized and valid data users who can satisfy the access policy so that it can be sent to CS for storage and sharing later. In order to increase security, it does not outsource a number of access policy attributes to Prime fog. The number should be as small as possible so that the minimum computational overhead is imposed on DO, and at the same time, Prime fog cannot decrypt the ciphertext without the need for DO.
DU: By using the Prime fog, DUs can access the relevant data in the cloud if they have the attributes accepted by the DO and the CS verifies it. Also, after accessing the information that is in the form of RNS, they use the CRT fog to convert it into binary.

4.2 System Framework

This scheme includes 4 phases, as explained below.

4.2.1 Phase 1 (Setup)

1. *AA_O({O}):* The AA takes the attribute set belonging to the DO from the input and calculates its dynamic range of the attributes *Dyn_O.* Then, it outputs and sends it to the DO.
2. *AA_U_Prime({U}):* The AA receives as input the attribute set *{U}* of DU. Subsequently, it calculates the value of each module with a length of 512 bits using the SHA3-512 function [29]. After that, it calculates the smallest prime number greater than the hash value for each module. Then it calculates the dynamic range of the prime modules *Dyn_U* and sends it to the output along with the corresponding set of prime numbers *{PU}.*

4.2.2 Phase 2 (KeyGen)

3. *AA_MO(Dyn_O):* The AA receives *Dyn_O* as input and subsequently outputs the unique random public key *MO* and sends it to the DO and the CS through a secure channel.
4. *AA_MU (Dyn_U)*: The AA receives *Dyn_U* as input and subsequently outputs the unique random private key *MU* based on the length of the plaintext *L (P_Text)* and the number of members of the *{PU}.* Then, sends it to the DU through a secure channel.

4.2.3 Phase 3 (Encryption)

This phase includes the functions of defining the access policy and converting the plaintext into sets of residues based on binary to RNS conversion or forward conversion.
5. *Fog_Prime ({O_Pol_P}):* The Prime fog receives a set of all attributes used in the access policy *{O_Pol_P}* except for a small number of attributes *{O_Pol_S}* that remain confidential to the DO to calculate their equivalent prime attribute modules. Each attribute module can be selected randomly from any attribute set in the access structure to remain confidential and not sent, provided that at least one attribute is selected from each attribute set. In RNS, if only one module or one residue related to a module is not specified, it is enough that it cannot be calculated uniquely during the reverse conversion of CRT. The access structure includes a number of separate attribute sets, and all the sets should not be fully available to the Prime fog so that it cannot decrypt the ciphertext by itself and without depending on the DO in case of intruder penetration. If an attribute is a member

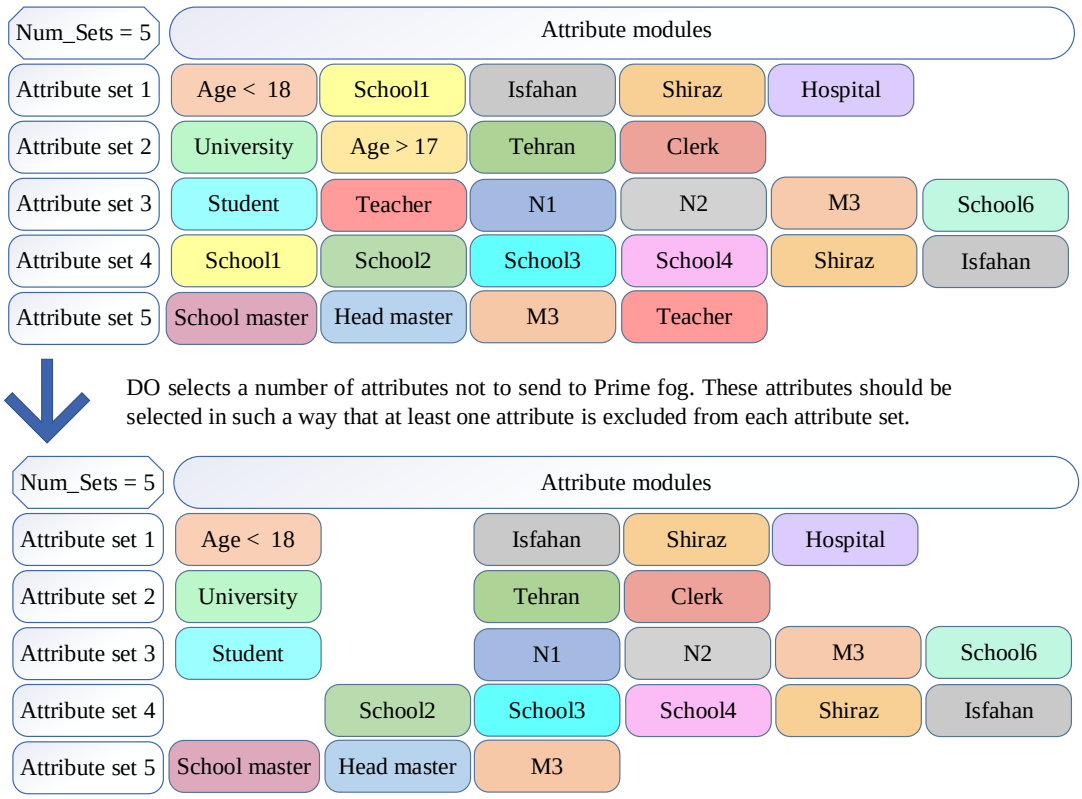

Fig. 3 How to select attributes that DO does not send to the Prime fog

in multiple attribute sets in the access structure, it is better not to send those attributes to the Prime fog to reduce the computational overhead in DO.

Please refer to Fig. 3 for better understanding. In this figure, it is assumed that *Num_Sets* = 5 and each attribute set contains an arbitrary number of attributes. The attributes *'school1', 'teacher', 'M3', 'Isfahan'*, and *'Shiraz'* are repeated in two attribute sets, and the rest of the attributes are only used in one set. DO arbitrarily selects '*school1', 'teacher',* and *'Age > 17'* attributes not to send to the Prime fog. So *{O_Pol_S} = {'school1', 'teacher', 'Age > 17'}* and *{O_Pol_P} = {'Age < 18', 'Isfahan', 'Shiraz', 'Hospital', 'University', 'Tehran', 'Clerk', 'Student', 'N1', 'N2', 'M3', 'School6', 'School2', 'School3', 'School4', 'Schoolmaster', 'Headmaster'}*. Another choice for *{O_Pol_S}* could be *'M3', 'Shiraz'* and *'Clerk'*. Because none of the attributes of attribute set 2 exist in the rest of attribute sets and are non-repetitive, DO can randomly choose one of them. Then, the hash value of each attribute sent is calculated by the Prime fog using the SHA3-512 function, and subsequently, the set of the smallest prime numbers greater than each of these hash values *{O_Pol_Fog}* is output and sent to the DO.

6. *DO_Prime ({O_Pol_S}):* The DO takes the set of secret attributes *{O_Pol_S}* that have not been sent to the Prime fog as input. Then it calculates the hash value of each of them using the SHA3-512 hash function. Subsequently, the set of the smallest prime numbers greater than each of these hash values *{O_Pol_DO}* is output.

7. *DO_Acc_Policy (Acc_Policy$_{MO}$):* The DO takes the set *Acc_Policy$_{MO}$ = ({O_Pol_DO}, {O_Pol_Fog})* as input and then selects and outputs the set of attribute sets $\{\{P\_Att_x\}\}_{1 \le x \le Num\_Sets}$. After that, it is sent to the AA via the secure channel.

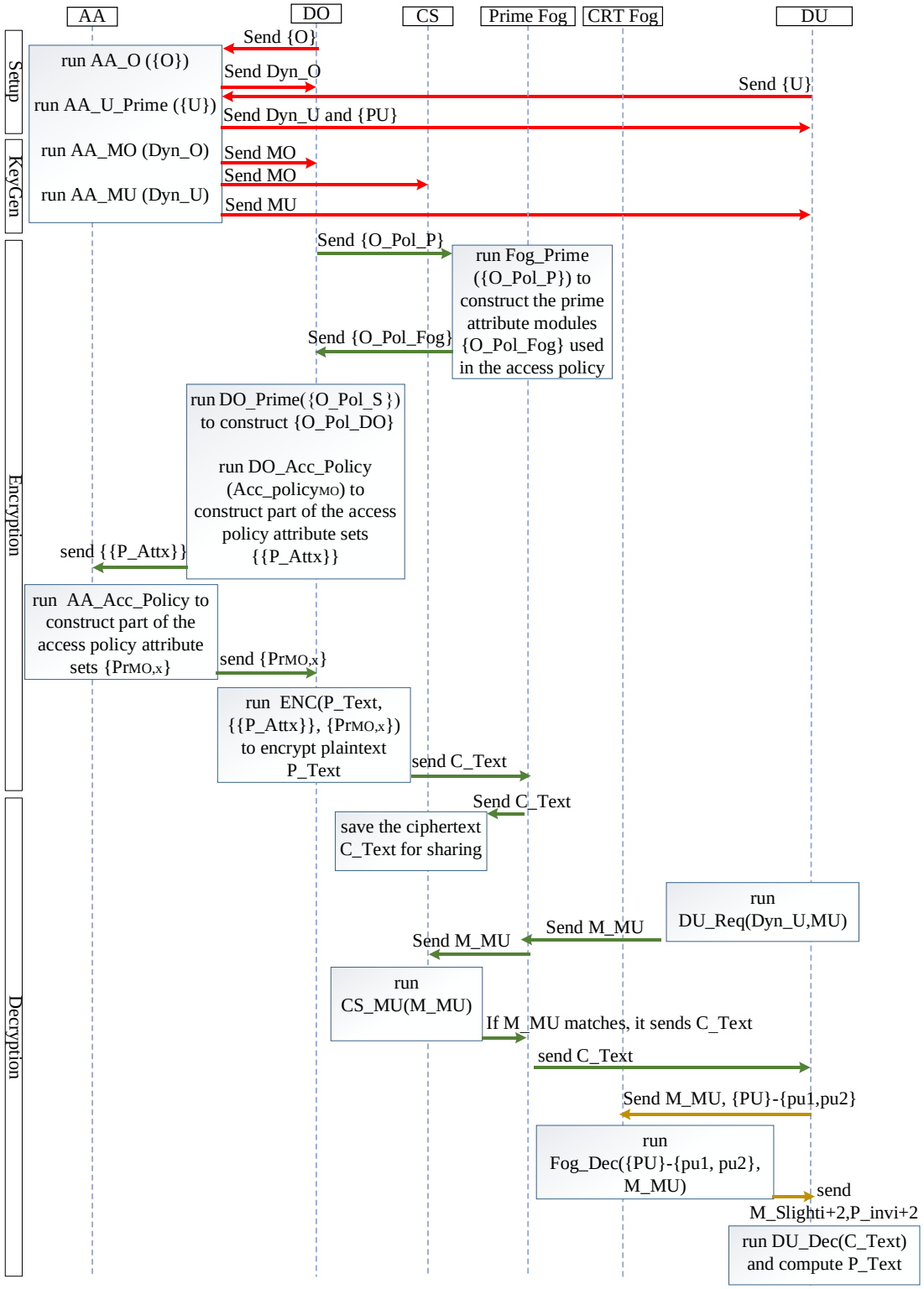

Fig. 4 Data flow of our scheme

8. *AA_Acc_Policy* ($\{\{P\_Att_x\}\}_{1\le x\le Num\_Sets}$): The AA takes the set $\{\{P\_Att_x\}\}_{1\le x\le Num\_Sets}$ as input and computes the dynamic range of each set of prime numbers $\{Dyn\_R_x\}_{1\le x\le Num\_Sets}$. Then, the set of random prime numbers $\{Pr_{MO,x}\}_{1\le x\le Num\_Sets}$ is selected based on the length of the plaintext *L(P_Text)* and the dynamic range of each prime number module set $L(Dyn\_R_x)$, and then it is output.

9. *ENC(P_Text,* $\{\{P\_Att_x\}\}_{1\le x\le Num\_Sets}$, $\{Pr_{MO,x}\}_{1\le x\le Num\_Sets}$): The DO takes the *P_Text,*

$\{\{P\_Att_x\}\}_{1\le x\le Num\_Sets}$, and $\{Pr_{MO,x}\}_{1\le x\le Num\_Sets}$ as inputs and outputs residue set of the *P_Text* concerning the $\{\{P\_Att_x\}\}_{1\le x\le Num\_Sets}$, and $\{Pr_{MO,x}\}_{1\le x\le Num\_Sets}$ prime modules as the ciphertext *C_Text.* Then *C_Text* is sent to the CS for data storage and sharing through the Prime fog.

4.2.4 Phase 4 (Decryption)

This phase contains functions that decrypt the *C_Text* using forward conversion or CRT-based reverse conversion to the *P_Text.*

10. *DU_Req(Dyn_U, MU):* The DU must send its request *M_MU* to the Prime fog and subsequently to the CS, so its prime attribute modules cannot be recognized.

11. *CS_MU(M_MU):* CS takes as input *M_MU* and compares it with all the $\{c'_x\}_{1\le x\le Num\_Sets}$. If, for an arbitrary j, matching between *M_MU* and c'_j is done, it sends c_j, $\tilde{c}_j$, and c'_j to the Prime fog. Accordingly, the Prime fog also sends it to the DU. Otherwise, it outputs $\perp$.

12. *Fog_Dec({PU} - {pu_1, pu_2}, M_MU):* Intending to reduce energy consumption and increase the speed of CRT calculations, the DU outsources the calculation of several multiplicative inverses used in the CRT equation to the CRT fog.

13. *DU_Dec (C_Text):* DU takes as input *C_Text* and outputs the *P_Text* With the help of the CRT equation.

The data flow of our scheme is shown in Fig. 4.

4.3 Security model

We assume that the CS, Prime fog, and CRT fog are semi-trusted, meaning they are doing their calculations correctly but maybe colluding. Accordingly, our security model covers the following properties.

(1) Confidentiality: Disclosure of data to unauthorized users is prevented.
(2) Resistance to collusion: Two or more users cannot access data that they cannot access individually by cooperating and sharing their private key.
(3) Fine-grained access control: Only DUs whose attributes match the access policy can access the ciphertext.
(4) Resistance to chosen-plaintext attack (CPA): The adversary should not be able to decrypt other ciphertexts by receiving several plaintexts and their identical ciphertexts.

5 Construction of the scheme

In many proposed schemes based on CP-ABE, by outsourcing part of the calculations to the cloud and fog entity, an attempt has been made to increase the processing speed and data storage capacity. The IoT network devices, which often have limited processing, storage, and energy resources, can provide the services needed by users more efficiently. In addition to these measures, algorithms need to be computationally simplified and have a parallel structure. One of the problems of CP-ABE-based schemes is the use of processes with high computational overhead, such as bilinear pairing and modular exponentiation. Another drawback is using fog computing sequentially. In other words, the data user must wait until the end of the fog processing. Considering the necessity of simplifying the parameters in CP-ABE schemes used in the IoT ecosystem and the presence of desirable features

in the RNS, such as lightweight and parallel structure and optimal efficiency in performing exponential operations, multiplication, and Addition because it converts huge integers into small integers, these features are used in this scheme. In this section, all the algorithms in the proposed RNS_ABE scheme are explained step by step and in detail.

5.1 Setup

1. *AA_O({O}): → Dyn_O*

Step 1: Each attribute of set $\{O\} = \{o_1, o_2, \ldots, o_n\}$ is defined as an expression and a string. For example, *Tehran hospital nurse* or *Full-time professor* or *Student of the university of Science and Research* is the attribute of the DO.

Step 2: The AA hashes each member of the set *{O}* by the SHA3-512 algorithm and puts its output in the set $\{PO\} = \{po_1, po_2, \ldots, po_n\}$.

Step 3: All members of set *{PO}* are multiplied by each other and considered as dynamic range $Dyn\_O = \prod_{po_i \in \{PO\}} po_i$ and subsequently sent to the DO through a secure channel.

2. *AA_U_Prime({U}) → {PU}, Dyn_U*

Step 1: Similar to the first step of the *AA_O({O})* algorithm, we run it for the set $\{U\} = \{u_1, u_2, \ldots, u_n\}$.

Step 2: The AA hashes each member of the set *{U}* by the SHA3-512 algorithm and puts its output in the set $\{HU\} = \{hu_1, hu_2, \ldots, hu_n\}$.

Step 3: The AA calculates the smallest prime number greater than the hash value of each of the DU attributes and loads it into the set $\{PU\} = \{pu_1, pu_2, \ldots, pu_n\}$.

Step 4: All members of set *{PU}* are multiplied by each other and considered as dynamic range $Dyn\_U = \prod_{pu_i \in \{PU\}} pu_i$ and subsequently sent to the DU through a secure channel.

5.2 Keygen

3. *AA_MO(Dyn_O) → MO*

Step 1: The AA outputs a unique random number with a length of 1024 bits as a public key *MO* and then sends it to DO and CS through a secure channel. *MO* does not need to be prime.

4. *AA_MU(Dyn_U) → MU*

Step 1: The AA must generate a unique random prime number *MU*, proportional to the length of the dynamic range *L(Dyn_U)* and the allowed length of the plaintext *L(P_Text)* produced by the DO according to the following relationship.

$$L(Dyn\_U) < L(P\_Text) < L(MU) + L(Dyn\_U)$$
$$L(MU) < L(P\_Text) \quad (4)$$

The first relation above guarantees that the output of the CRT reverse function is a unique number, and the second relation guarantees that the CRT reverse function must be executed to calculate plaintext, and there is no possibility of bypassing this function and not depending on it. In other words, the above two relations determine the minimum and maximum length of the plaintext according to the size of the prime attribute modules and the size of the private key *MU*. Then sends it to the DU through a secure channel.

5.3 Encryption

In general, in CP-ABE schemes, the structure of the access policy is based on the access tree or linear secret sharing scheme (LSSS). The access tree has a high expression, but its efficiency is not optimal due to recursive functions and exponentiation. However, instead, in LSSS, the expression has decreased, but the applicability and efficiency have been improved. Therefore, algorithms based on LSSS, such as [27,28], were presented to improve its efficiency and expressiveness. However, both models of access policies have an integrated structure, and it is not possible to separate them into independent sub-parts that can help simplify and parallelize calculations.

On the other hand, revoking attributes or revoking users' access has a high computational overhead, and in many cases, the entire access policy needs to be updated. Therefore, in this research, the structure of the access policy was proposed based on the CRT secret sharing scheme [30] for each valid and satisfactory attribute set. Due to the semi-reliability of the CS and the Prime fog, instead of placing prime attribute sets in the access policy, only the dynamic range of each of those sets is loaded into it.

5. *Fog_Prime({O_Pol_P}) → {O_Pol_Fog}*

Step 1: The DO sends the set of all attributes used in the access policy *{O_Pol_P},* except for a limited number, to the Prime fog. This improves security and privacy.

Step 2: The Prime fog hashes each member of the set *{O_Pol_P}* by the SHA3-512 algorithm and puts its output in the set *{O_Pol_H1}*.

Step 3: The Prime fog calculates the smallest prime number greater than the hash value of each of the *{O_Pol_H1}* attributes and load it into the set *{O_Pol_Fog}*.

Step 4: The set *{O_Pol_Fog}* is sent to the DO.

6. DO_Prime ({O_Pol_S}) → {O_Pol_DO}

Step 1: The DO hashes each set member *{O_Pol_S}* by the SHA3-512 algorithm and puts its output in the set *{O_Pol_H2}*.

Step 2: The DO calculates the smallest prime number greater than the hash value of each of the *{O_Pol_H2}* attributes and loads it into the set *{O_Pol_DO}*.

7. *DO_Acc_Policy (Acc_Policy$_{MO}$) → {{P_Att$_x$}}$_{1\le x\le Num\_Sets}$*

Step 1: The DO specifies the number of sets of primes *Num_Sets* that are equivalent to the attributes the access policy is based on.

Step 2: The DO designs and chooses a set *{{P_Att$_x$}}$_{1\le x\le Num\_Sets}$* that includes the sets of attribute modules that make up the access policy. The attribute set belonging to the DU must exactly match one of these sets in order to satisfy the access policy and thus be able to decrypt the ciphertext *C_Text* using RNS to binary conversion.

Step 3: It sends *{{P_Att$_x$}}$_{1\le x\le Num\_Sets}$* to the AA.

8. *AA_Acc_Policy({{P_Att$_x$}}$_{1\le x\le Num\_Sets}$) → {Pr$_{MO,x}$}$_{1\le x\le Num\_Sets}$*

Step 1: The AA calculates the dynamic range of each attribute set *{Dyn_R$_x$}$_{1\le x\le Num\_Sets}$* in *{{P_Att$_x$}}$_{1\le x\le Num\_Sets}$*.

Step 2: Based on the following relationship, unique prime numbers *{Pr$_{MO,x}$}$_{1\le x\le Num\_Sets}$* are generated and output.

For $x = 1$ to *Num_Sets*:

$$L(Dyn\_R_x) < L(P\_Text) < L(Pr_{MO,x}) + L(Dyn\_R_x)$$
$$L(Pr_{MO,x}) < L(P\_Text) \quad (5)$$

Step 3: The AA sends the unique prime numbers *{Pr$_{MO,x}$}$_{1\le x\le Num\_Sets}$* to the DO.

9. *ENC(P_Text, {{P_Att$_x$}}$_{1\le x\le Num\_Sets}$, {Pr$_{MO,x}$}$_{1\le x\le Num\_Sets}$) → C_Text*

Step 1: The set of plaintext *P_Text* residues corresponding to the set of modules $\{Pr_{MO,x}\}_{1\le x\le Num\_Sets}$ and $\{\{P\_Att_x\}\}_{1\le x\le Num\_Sets}$, which are prime numbers, is calculated

$$C\_Text = \begin{pmatrix} \{\{c_x\} = P\_Text \bmod \{P\_Att_x\}\}_{1\le x\le Num\_Sets} \\ \{\tilde{c}_x = P\_Text \bmod pr_{MO,x}\}_{1\le x\le Num\_Sets} \\ \{\acute{c}_x = Dyn\_R_x \times pr_{MO,x}\}_{1\le x\le Num\_Sets} \end{pmatrix} \quad (6)$$

Step 2: The *C_Text* is sent to the Prime fog and stored there for a short time.

Step 3: The *C_Text* is sent from the Prime fog to the CS and stored there for a long time.

5.4 Decryption

10. *DU_Req(Dyn_U, MU)* → *M_MU*

Step 1: The DU calculates $M\_MU = Dyn\_U \times MU$.

Step 2: It sends to the Prime fog. Later, the Prime fog also sends *M_MU* to the CS.

11. *CS_MU(M_MU)* → $(\{c_j\}, \tilde{c}_j, c'_j)$ or $\perp$.

It is fully explained in the System Framework section.

12. *Fog_Dec({PU}-{pu1,pu2}, M_MU)* → $\{M\_Slight_{i+2}\}_{1\le j\le n-2}$, $\{p\_inv_{i+2}\}_{1\le j\le n-2}$

Step 1: The set of prime attribute modules of the DU, except for two prime attribute modules, is received by the CRT fog. The number of DU's attribute modules *n* is equal to the number of the set $\{c_j\}$. The number *n+1* refers to the additional module or the *MU* private key.

Step 2: The CRT fog computes

$$\{M\_Slight_{i+2}\}_{1\le i\le n-2} = \frac{M\_MU}{pu_{i+2}} \; for\; 1 \le i \le n-2 \quad (7)$$

Step 3: The CRT fog computes

$$\{p\_inv_{i+2} \mid (p\_inv_{i+2} \times M\_Slight_{i+2}) \bmod pu_{i+2} = 1\}_{1\le j\le n-2} \quad for\ 1 \le i \le n\text{-}2 \quad (8)$$

Step 4: The CRT fog sends $\{M\_Slight_{i+2}\}_{1\le j\le n-2}$ and $\{p\_inv_{i+2}\}_{1\le j\le n-2}$ to the DU.

13. *DU_Dec(C_Text)* → *P_Text*

Step 1: The DU computes

$$\{M\_Slight_i\}_{1\le i\le 2} = \frac{M\_MU}{pu_i} \; for\; 1 \le i \le 2 \quad (9)$$

Step 2: The DU computes

$$\{p\_inv_i \mid (p\_inv_i \times M\_Slight_i) \bmod pu_i = 1\}_{1\le j\le 2} \quad for\ 1 \le i \le 2 \quad (10)$$

Step 3: The DU computes p_inv_{n+1} according to the following relationship.

$$M\_Slight_{n+1} = \frac{M\_MU}{MU} = Dyn\_U$$

$$(p\_inv_{n+1} \times Dyn\_U) \bmod MU = 1 \quad (11)$$

Step 4: The DU computes *P_Text* based on the CRT and according to the following equation. $c_{j,i}$ means the *ith* member of $\{c_j\}$ and $c_{j,n+1} = \tilde{c}_j$.

$$P\_Text = \sum_{i=1}^{n+1} c_{j,i} \times M\_Slight_i \times p\_inv_i \bmod M\_MU \quad (12)$$

6 Security Analysis

The security of CP-ABE schemes, which is based on bilinear pairing, is proven chiefly based on the decisional bilinear Diffie-Hellman (DBDH) assumptions in the Random Oracle Model and based on a 5-step game [15-17, 20, 31]. In this game, it is proved that if the DBDH assumption is valid,

the adversary does not have a significant chance to distinguish the correct ciphertext from two ciphertexts. Therefore, the scheme's security against CPA attacks is guaranteed. The limitation of the Oracle Random model is that it is an ideal model and has not yet been implemented in practice, so a good hash function like SHA3 can be used instead [32].

In the phase of encryption, the security of the RNS-ABE scheme is guaranteed based on the non-factorization of the product of two large prime numbers. In the phase of decryption, it is guaranteed based on the security of the CRT-based secret-sharing scheme [33]. An efficient algorithm that can factorize a huge composite number consisting only of the product of large prime numbers is unknown. A slightly simpler example of this composite number, if it consists of only two large prime numbers, is semi-prime number. When two prime numbers are very large and randomly chosen and have a value relatively close to each other, even the fastest algorithms on the fastest computers cannot analyze them efficiently. One of the appropriate criteria to understand this issue is paying attention to RSA numbers. The maximum RSA-250 (829 bits) with a processing cost of 2700 core/years has been factorized [34]. It is estimated that factoring a 1024-bit key is about 200 times harder than RSA-250, or about 500,000 core/years [35]. Currently, the most efficient algorithm for the factorization of semi-prime numbers is called the General Number Field Sieve (GNFS) [36]. The cost of factoring the semi-prime number N with GNFS is equal to $Exp\left((64/9)^{1/3}\ (log\ N)^{1/3}(log\ log\ N)^{2/3}\ \left(1+o(1)\right)\right)$ [36]. This running time is sub-exponential, meaning it is faster than a polynomial time algorithm but slower than an exponential time algorithm. The security of all currently popular public-key cryptographic methods is based on the problems in number theory: 1- Semi-prime factorization, such as RSA encryption. 2- Discrete logarithms in finite fields such as Diffie-Hellman key exchange and 3- discrete logarithms on elliptic curves such as ECC and bilinear pairing [35]. Due to the widespread use of RSA, attacks have been implemented to break it, which can be successful for two reasons. First, small prime numbers p and q are selected. Secondly, small secret key d is selected. Based on the continued fractions algorithm and the Coppersmith method, and lattice reduction methods, the minimum allowed value of d has been improved to resist attacks [37]. The security of our scheme during binary to RNS conversion, unlike the RSA encryption algorithm, is based only on the difficulty of factoring into large prime factors, so the methods based on continued fractions cannot damage the proposed scheme. In Theorem 2 of [30], the conditions for producing a unique number within the allowed range based on CRT are specified. Therefore, we analyze the RNS-ABE scheme based on the security features of confidentiality, resistance to collusion, fine-grained access control, and resistance to CPA attacks.

6.1 Confidentiality

In this research, AA is considered the only wholly reliable entity. Due to the conversion of the plaintext by the DO into several remaining sets and the lack of knowledge of the CS, Prime fog, and unauthorized DU entities from each of the modules in the set of valid attribute modules in the access policy, the content of the plaintext, in general, it is unknown. According to the CRT equation, it is impossible to calculate the plaintext without calculating the value of modules. There are three possible modes for discovering modules.

Mode 1: The product of the modules should be factorized into its prime factors. As mentioned earlier, our security assumption is based on the difficulty of factoring a number into its large prime factors. The basis of RSA cryptographic security is also based on this. In RSA encryption, the public key N consists of the product of two large prime numbers, p and q. If the number of prime factors exceeds

two factors, provided that all of them are greater than RSA-1024, this product is still secure.
Mode 2: A DU should have valid and satisfying attributes of the access structure.
Mode 3: The Prime fog knows about most of the prime modules, but firstly, it does not know at least one prime module of each attribute set in the access policy. Secondly, it does not know the unique random numbers $\{Pr_{MO,x}\}_{1\leq x\leq Num\_Sets}$. Thus, practically with the remaining set sent to it cannot detect *P_Text*. The CRT fog is an entity that is aware of several valid modules. When sending data to it, several measures have been considered as follows so that the scheme's security is not compromised.

- It is assumed that there is no connection between CRT fog and CS and Prime fog. It should be located in another zone to ensure this.
- Despite the communication restriction measures, if the CRT fog is compromised or colluding with an unauthorized user, the DU must not send at least two larger modules to the CRT. To improve security, fewer modules can be sent to the CRT fog if the DU's hardware and power supplies are capable enough. For more security, a secure channel can be used to send prime modules to the CRT fog.
- The DU should not send any data from the set of residuals received from the Prime fog to the CRT fog. If all the attribute modules of an authorized data user have been discovered by the adversary or colluded by the CRT fog with an unauthorized entity, but due to the lack of knowledge of the remaining set received by the DU, they can't discover the plaintext based on the CRT conversion.

It is worth mentioning that in this research, the reverse conversion method is considered a CRT method due to its parallel structure, low delay, and less hardware area [26]. In this method, the value of the following variables must be known to achieve a unique output.

a) All the prime modules in the attribute set
b) All residues in the residue set corresponding to each prime module
c) All multiplicative inverses belonging to each prime module

Therefore, if any of the above cases are not known, the security of this scheme will be maintained. In other words, in mode 1, the variables related to a) and c) are hidden. In mode 2, variables related to a), b), and c) are hidden for unauthorized users, and in mode 3, at least two modules related to a) and c) are hidden, and b) are completely hidden.

6.2 Resistance to collusion

The requirement for collusion between two entities is the possibility of combining their set of prime modules and their set of residues in order to be able to access the plaintext that each of them cannot discover on their own. Considering that for each valid DU in the access policy, a unique module is issued by the AA as the private key *MU*, and so collusion between two or more users is not possible. For a better understanding, pay attention to the following example.

6.2.1 Example

P_Text = 12345678.
The set of prime modules of the authorized data user is {7, 11, 19, 23, 29} and *MU*= 37.
The set of prime modules of the first unauthorized user is {7, 11, 19} and *MU*=719.
The set of prime number modules of the second unauthorized user is {23, 29} and *MU*=773.
The set of residuals authorized data user stored in the CS is {2,4,10,14,1,36}.
The set of residuals for the first unauthorized data user is {2,4,10,448}.
The set of residuals for the Second Unauthorized Data User is {14,1,95}.

If collusion takes place due to not knowing the value of MU = 37, two unauthorized users with residue 448 and residue 95 cannot calculate the correct plaintext. Therefore, this scheme is resistant to collusion.

6.3 Fine-grained access control

According to the proposed scheme, only DUs can access the plaintext that has valid attributes and can satisfy the access policy. Due to the simple and flexible structure of the access policy and not using complex and high-overhead calculations such as modular exponentiation and bilinear pairing mapping, as well as the possibility of converting the structure of the access policy into independent components, the flexibility of the scheme during RNS to binary conversion has been improved.

6.4 Resistance to Chosen Plaintext Attack (CPA):

In this research, the game model is considered based on RNS calculations and assumes that it is difficult to factorize a number into its large prime factors by sampling the game of [20,29]; defined according to the following method. This game consists of two entities: the adversary and the challenger. If the probability of detecting the plaintext belonging to the ciphertext received by the adversary is significantly higher than ½, we consider it a significant score, and it means the adversary wins the game. Otherwise, the security of the scheme is guaranteed against the CPA.

6.4.1 CPA game

1. Pre-Initialization: $Acc\_Policy_{Adversary}$ is sent by the adversary to the challenger.
2. Setup: The adversary's *MO*, the allowed length of the plaintext, and the limits that must be observed by the adversary are sent to him by the challenger.
3. Phase 1: The adversary sends the challenger sets of attributes that do not satisfy an access policy, and then the challenger selects a unique random *MU* key for each of them and sends it to the adversary.
4. Challenge: The adversary sends two plaintexts, *P_Text0* and *P_Text1,* which have the same length to the challenger. The challenger flips a coin. If it is heads, the *P_Text0* is selected; otherwise, the *P_Text1* is selected. So, based on the modules in $Acc\_Policy_{Adversary}$, the challenger converts the plaintext from binary to RNS and then sends it to the adversary.
5. Phase 2: Again, the adversary acts like phase 1.
6. Guess: Based on the score ξ the adversary can guess which plaintext belongs to the received ciphertext. If $\xi \cong 0$ the adversary cannot obtain useful information, it is concluded that no significant points have been given to the
adversary, and so our scheme is secure.

According to phase 1, the challenger must not send $\{pr_{MO,x} | x \epsilon Num\_Sets\}$ to the adversary. Therefore, the adversary cannot calculate the plaintext by using the CRT conversion despite having sets of residuals. The maximum score of the adversary is equal to the probability of correctly choosing a module of a certain length. So: $\xi = \frac{1}{min\{pr_{MO,x}\}_{1 \le x \le Num\_Sets}} \cong 0$, and as a result, our proposed scheme is secure.

7. Performance analysis

In this section, we first examine the effectiveness of the
proposed scheme theoretically and estimate its time cost based on the leading operators, and compare it with three articles [14,17,18]. Then, we will simulate and compare the time cost of the proposed scheme with [14,17,18] whose underlying calculations are based on bilinear pairing in

Table 1. The symbols used in the performance analysis

Symbol	Description	Symbol	Description
Keygen	Private key generation	Prime	time cost of a next prime number function
Enc	Encryption	SHA3	SHA3-512bit hash function
Dec	Decryption	∑	$\sum_{x \in non-leaf\ node} K_x E_p$ K_x is equivalent to the threshold value
Fog	Fog entity		
DU	Data user entity		
DO	Data owner entity		
AA	Attribute authority	Rnd	Choosing a random integer with a certain length
CA	Central authority entity		
S	Number of attributes in private key		
Y	Number of attributes in access policy	Mul	Multiplication operation
Numset	Number of sets in access policy	Div	Division operation
E_p	time cost of an exponent operation in G_T	Mod	Residue operation
E_0	time cost of an exponent operation in G0	Inv	Multiplicative inverse operation
P	time cost of a bilinear pairing operation	Sym	Symetric Encryption or Decryption

Table 2. Computational cost

Schemes	Keygen		Enc			Dec	
[14]	AA→DU	AA→Fog	Fog		DO	Fog	DU
	E_0	$(2S+4)E_0$ +Smul	$(2Y+2)E_0$		E_p +3.E_0 +3.mul +Sym	(2S+2)P+∑+ (2S+2)div	P +mul +div +Sym
[17]	AA→DU	CA→DU	Fog	AA	DO	Fog	DU
	2.S.E_0	$(2S+3)E_0$ +(S+1)mul	$(3Y+2)E_0$ + Y. mul	YE_0	E_p +3.E_0+mul	(3S+2)P +(S+1)mul +(S+1)div+∑	P +mul +div
[18]	AA→DU	AA→Fog	Fog		DO	Fog	DU
	$(S+1)E_0$	$(2S+4)E_0$	$(Y+2)E_0$		E_p +2.E_0 +3.mul	(S+2)P +2div +∑	P +mul +div
Ours	AA→DU	AA→DO	Fog	AA	DO	Fog	DU
	Prime	rnd	(Y-Numset) Prime +(Y- Numset) SHA3	Y. mul+ Numset. Prime	(Y+ Numset) mod +Numset.mul	(S-2)div +(S-2)inv	3.div +3.inv +3.(S+1)mul +1.mod

three phases, Keygen, encryption, and decryption.

7.1 Theoretical analysis

All symbols used in theoretical performance analysis are shown in Table 1. In this table, all the operations that have a computational cost are introduced. Symbols marked in green can be ignored if entities have high processing speed. Computational overhead is based on the basic operations introduced in Table 2. According to the analysis, for the operation that is colored green in Table 1, the computation time is less than one millisecond on a single-core 1.73 GHz CPU, and therefore it can be ignored. For example, the time spent to perform the RC4 stream symmetric encryption operation with a key length of 256 bits and a plaintext of 1024 characters is about 0.5 ms. In the

proposed scheme, unlike the other three articles, the SHA3 practical hash function with a digest length of 512 bits is used instead of the Random Oracle Model. If the ideal Oracle model is used, the SHA3 computation overhead should be removed from the encryption part of the proposed scheme.

7.2 Experimental analysis

We perform the simulation tests on a laptop with a Core i7 Q740 CPU, frequency 1.73 GHz, 3 GB memory, virtual machine Oracle VM VirtualBox version 5.2.38, and Ubuntu 18.04 LTS. We considered the hardware of the IoT devices to be a single-core CPU and 512 MB of memory and the fog computing to be a six-core CPU and 2 GB of memory. We generated A-type pairing parameters, which is the fastest pairing map, with 160-bit group order in a finite field of 512 bits order [38]. The number of attributes for encryption used in the tests for encryption is from 5 to 60 and for decryption is from 5 to 61, and the result of the test is the average number of 20 runs. In this test, we calculate the time cost of the three phases of Keygen, encryption, and decryption of the proposed scheme with C code and compare it with the three articles [14,17,18]. We also assume that the access tree in these three articles has three children per non-leaf node. Accordingly, if it has $(n-1)/2$ non-leaf nodes, it will have n leaf nodes.

Fig.5 compares the time cost of the proposed scheme at the keygen phase with three articles [14,17,18]. Because only two *MO* and *MU* keys are produced, this scheme, unlike the other three schemes, is not dependent on the number of attributes but depends on the difference in plaintext length and *Dyn_U* length. The ideal state is when this difference is exactly equal to the length of an attribute prime module, which is 512 bits here. In this figure, we calculated the time cost of the keygen phase for 3 *MU* secret keys with different lengths *L(MU)* of 512, 1024, and 1536 bits, which are displayed respectively according to the tags RNS-ABE_Mu_512, RNS-ABE_Mu_1024, and RNS-ABE_Mu_1536 for different number of attributes. According to the simulation results, if this difference reaches more than three times the length of the attribute prime module, the cost of the Keygen is unacceptable. In Fig.6, the data owner's encryption time is compared. The proposed scheme is assumed to increase access policy sets as the number of attributes increases, and for every five attributes, one set has been added to the access policy. Unlike the other three comparative articles [14,17,18], the time cost of the proposed scheme has a linear relationship with the number of attributes. In Fig.7, the time cost of outsourced encryption calculations is compared. In the proposed scheme, similar to [17], a part of the encryption calculations is outsourced to the fog computing and another part is outsourced to the AA entity. Articles [14,18] only outsource the encryption phase to the fog. The proposed scheme assumes that the SHA3-hash function with a 512-bit digest length is used, which is reduced if the Random Oracle Model is used instead. In Fig.8, the time cost of outsourced decryption is calculated. The most important feature of this scheme is a very low time cost of decryption compared to the other three schemes. Fig.9 compares the time cost of the DU, which has very low overhead for the proposed scheme compared to the other three schemes. Fig.10 shows the time cost for the total calculation of decryption and encryption. As can be seen, due to not using recursive functions and bilinear pairing, the time cost of the proposed RNS-ABE scheme has been significantly improved. The biggest time cost for this scheme is related to calculating the prime number larger than the hash value, which can be focused on its improvement in the future.

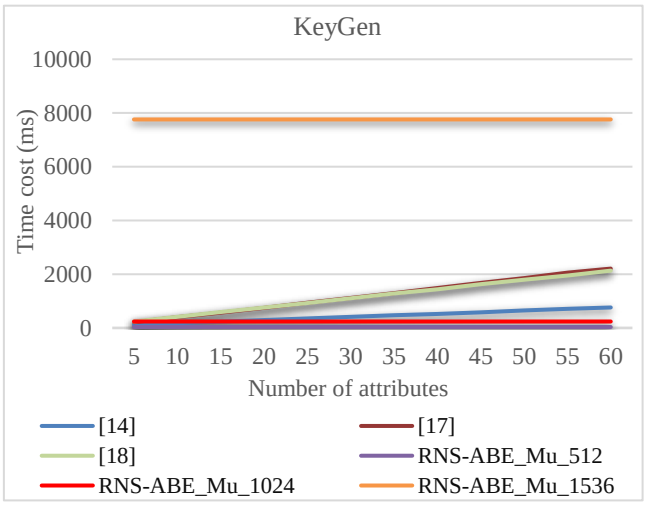

Fig. 5 Comparison of time cost for Keygen.

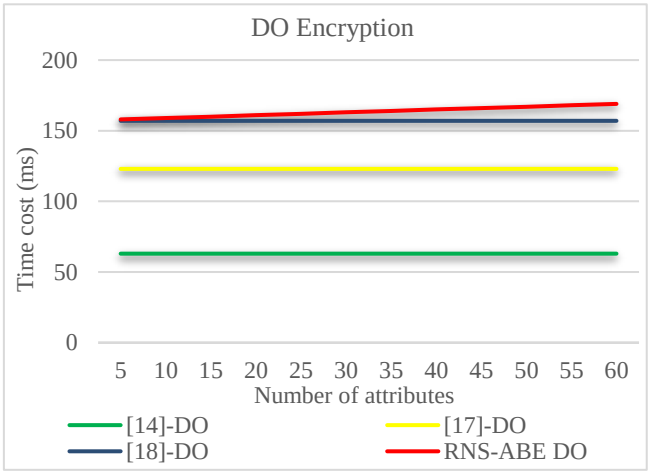

Fig. 6 Comparison of time cost for encryption by DO.

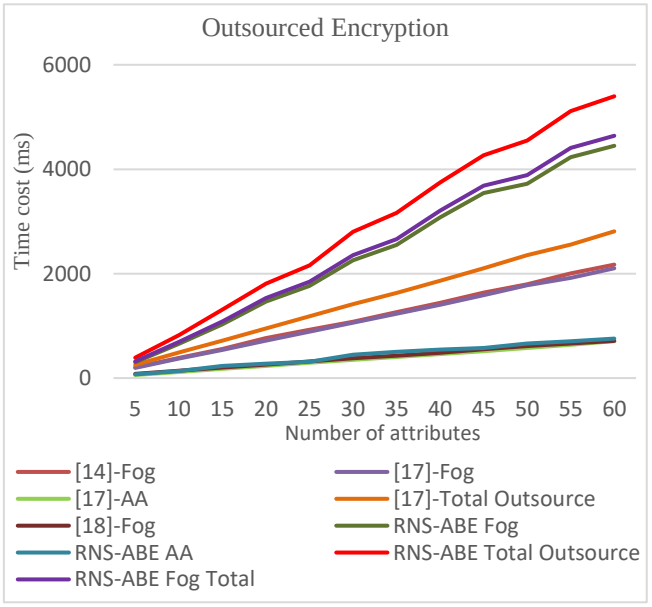

Fig. 7 Comparison of time cost for encryption by fog or AA.

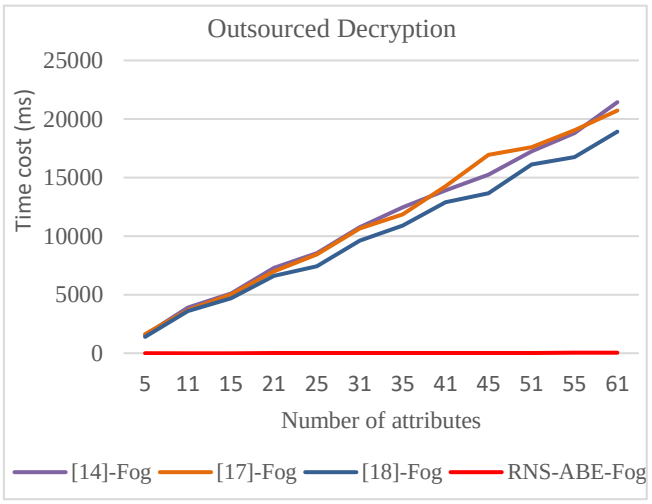

Fig. 8 Comparison of time cost for decryption by fog.

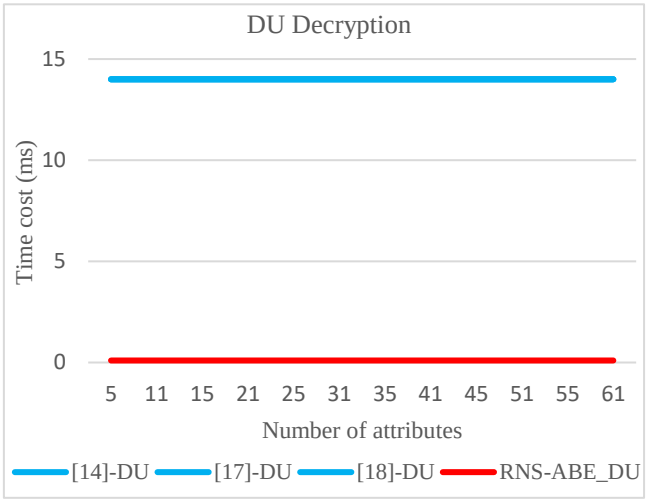

Fig. 9 Comparison of time cost for decryption by DU.

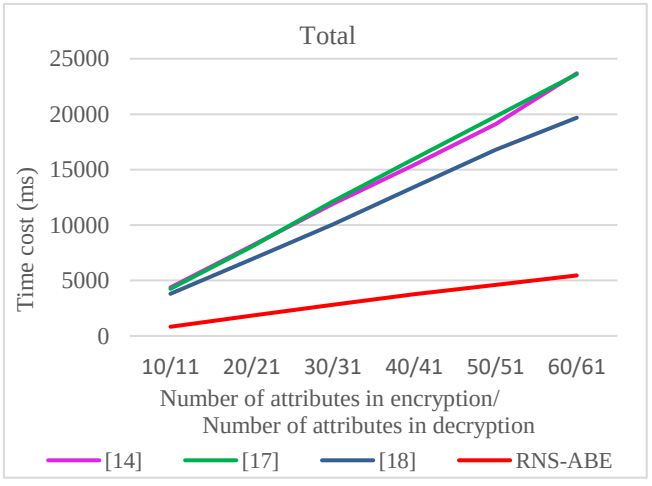

Fig. 10 Comparison of time cost for total encryption and decryption.

8 Conclusion

In this research, we proposed a fine-grained, lightweight, and fast access control scheme based on

ABE and RNS properties. The access structure in this scheme is new and agile, and unlike the access tree, it has independent subsections. Therefore, it can be accelerated by using parallel computing. We tried as much as possible so that all operational details were considered in the time cost analysis. The architecture of this scheme has two types of fog computing infrastructure, one for calculations of the smallest prime number greater than the hash value and the other for inverse multiplicative calculations for prime modules. Based on the simulation of the time cost, the average sum of encryption and decryption has decreased in both cases, assuming that the fog computing is ideal and its time cost is zero, similar to [14], and secondly, assuming that it is not ideal. In this scheme, if we use the Random Oracle Model instead of SHA3 with a digest length of 512 bits, the time cost can be reduced. In the future, we plan to complete this scheme and add the possibility of revoking a user or attributes and keyword search.

Statements and Declarations

Funding

The authors did not receive support from any organization for the submitted work.

Competing interests

The authors have no relevant financial or non-financial interests to disclose.

Author Contributions

All authors have contributed equally.

Data Availability

All data generated or analyzed during this study are available from the corresponding author upon reasonable request.

References

[1] Hassija, V., Chamola, V., Saxena, V., Jain, D., Goyal, P., & Sikdar, B.: A survey on IoT security: application areas, security threats, and solution architectures. IEEE Access, 7, 82721-82743 (2019). https://doi.org/10.1109/ACCESS.2019.2924045.

[2] Ahmed, H. I., Nasr, A. A., Abdel-Mageid, S., & Aslan, H. K.: A survey of IoT security threats and defenses. International Journal of Advanced Computer Research, 9(45), 325-350 (2019). https://dx.doi.org/10.19101/IJACR.2019.940088.

[3] Messaoud, S., Bradai, A., Bukhari, S. H. R., Quang, P. T. A., Ahmed, O. B., & Atri, M.: A survey on machine learning in internet of things: algorithms, strategies, and applications. Internet of Things, 12, 100314 (2020). https://doi.org/10.1016/j.iot.2020. 100177.

[4] Aazam, M., Zeadally, S., & Harras, K. A.: Fog computing architecture, evaluation, and future research directions. IEEE Communications Magazine, 56(5), 46-52 (2018). https://doi.org/10.1109/MCOM.2018.1700707.

[5] Aleisa, M. A., Abuhussein, A., & Sheldon, F. T.: Access control in fog computing: Challenges and research agenda. IEEE Access, 8, 83986-83999 (2020). https://doi.org/10.1109/ACCESS.2020. 2992460.

[6] Zhang, P., Liu, J. K., Yu, F. R., Sookhak, M., Au, M. H., & Luo, X.: A survey on access control in fog computing. IEEE Communications Magazine, 56(2), 144-149 (2018). https://doi.org/10.1109/ MCOM.2018.1700333.

[7] Ali, M., Sadeghi, M. R., & Liu, X.: Lightweight revocable hierarchical attribute-based encryption for internet of things. IEEE Access, 8, 23951-23964 (2020).

https://doi.org/10.1109/ACCESS. 2020.2969957.
[8] Mohan, P.V.A.: Residue number systems: Theory and applications. Basel, Birghauser, pp. 1-8 (2016). https://doi.org/10.1007/978-3-319-41385-3
[9] Bethencourt, J., Sahai, A., & Waters, B.: 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20–23 May 2007, Oakland, California, USA (2007). https://doi.org/10.1109/SP.2007.11.
[10] Oualha, N., & Nguyen, K. T.: Lightweight attribute-based encryption for the internet of things. In 2016 25th International Conference on Computer Communication and Networks (ICCCN) pp. 1-6 (2016, August). https://doi.org/10.1109/ICCCN.2016.7568538.
[11] He, H., Zhang, J., Gu, J., Hu, Y., & Xu, F.: A fine-grained and lightweight data access control scheme for WSN-integrated cloud computing. Cluster Computing, 20, 1457-1472 (2017). https://doi.org/10.1007/s10586-017-0863-y.
[12] Zhang, P., Chen, Z., Liu, J. K., Liang, K., & Liu, H.: An efficient access control scheme with outsourcing capability and attribute update for fog computing. Future Generation Computer Systems, 78, 753-762 (2018). https://doi.org/10.1016/j.future.2016.12.015.
[13] Huang, Q., Yang, Y., & Shen, M.: Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing. Future Generation Computer Systems, 72, 239-249 (2017). https://doi.org/10.1016/j.future.2016.09.021.
[14] Huang, Q., Yang, Y., & Wang, L.: Secure data access control with ciphertext update and computation outsourcing in fog computing for Internet of Things. IEEE Access, 5, 12941-12950 (2017). https://doi.org/10.1109/ACCESS.2017.2727054.
[15] Amor, A. B., Abid, M., & Meddeb, A.: Secure fog-based e-learning scheme. IEEE Access, 8, 31920-31933 (2020). https://doi.org/10.1109/ACCESS.2020.2973325.
[16] Xu, S., Ning, J., Li, Y., Zhang, Y., Xu, G., Huang, X., & Deng, R. H.: Match in my way: Fine-grained bilateral access control for secure cloud-fog computing. IEEE Transactions on Dependable and Secure Computing, 19(2), 1064-1077 (2020). https://doi.org/10.1109/TDSC.2020.3001557.
[17] Li, L., Wang, Z., & Li, N.: Efficient attribute-based encryption outsourcing scheme with user and attribute revocation for fog-enabled IoT. IEEE Access, 8, 176738-176749 (2020). https://doi.org/10.1109/ACCESS.2020.3025140.
[18] Miao, Y., Ma, J., Liu, X., Weng, J., Li, H., & Li, H.: Lightweight fine-grained search over encrypted data in fog computing. IEEE Transactions on Services Computing, 12(5), 772-785 (2018). https://doi.org/10.1109/TSC.2018.2823309.
[19] Zhang, J., Cheng, Z., Cheng, X., & Chen, B.: OAC-HAS: outsourced access control with hidden access structures in fog-enhanced IoT systems. Connection Science, 33(4), 1060-1076 (2021). https://doi.org/10.1080/09540091.2020.1841096.
[20] Feng, C., Yu, K., Aloqaily, M., Alazab, M., Lv, Z., & Mumtaz, S.: Attribute-based encryption with parallel outsourced decryption for edge intelligent IoV. IEEE Transactions on Vehicular Technology, 69(11), 13784-13795 (2020). https://doi.org/10.1109/TVT.2020.3027568.
[21] Khashan, O. A.: Hybrid lightweight proxy re-encryption scheme for secure fog-to-things environment. IEEE Access, 8, 66878-66887 (2020). https://doi.org/10.1109/ACCESS.2020. 2984317.
[22] Zhang, A., Wang, X., Ye, X., & Xie, X.: Lightweight and fine-grained access control for cloud–fog-based electronic medical record sharing systems. International Journal of Communication Systems, 34(13), e4909 (2021). https://doi.org/10.1002/dac.4909.
[23] Tu, Y., Yang, G., Wang, J., & Su, Q.: A secure, efficient and verifiable multimedia data sharing

scheme in fog networking system. Cluster Computing, 24(1), 225-247 (2021). https://doi.org/10.1007/s10586-020-03101-6.
[24] Saidi, A., Nouali, O., & Amira, A.: SHARE-ABE: an efficient and secure data sharing framework based on ciphertext-policy attribute-based encryption and Fog computing. Cluster Computing, 25(1), 167-185 (2022). https://doi.org/10.1007/s10586-021-03382-5.
[25] Aghili, S. F., Sedaghat, M., Singelée, D., & Gupta, M.: MLS-ABAC: Efficient Multi-Level Security Attribute-Based Access Control scheme. Future Generation Computer Systems, 131, 75-90 (2022). https://doi.org/10.1016/j.future.2022.01.003.
[26] Mohan, P.V.A.: Residue number systems: Theory and applications, Basel, Birghauser, pp. 27-128 (2016). https://doi.org/10.1007/978-3-319-41385-3
[27] Liu, Z., Cao, Z., & Wong, D. S.: Efficient generation of linear secret sharing scheme matrices from threshold access trees. Cryptology ePrint Archive (2010).
[28] Lewko, A., & Waters, B.: Decentralizing attribute-based encryption. In Annual international conference on the theory and applications of cryptographic techniques (pp. 568-588). Springer, Berlin, Heidelberg (2011, May). https://doi.org/10.1007/978-3-642-20465-4_31.
[29] Paar, C., & Pelzl, J.: Sha-3 and the hash function keccak. Understanding Cryptography-A Textbook for Students and Practitioners (2010).
[30] Wu, L., Miao, F., Meng, K., & Wang, X.: A simple construction of CRT-based ideal secret sharing scheme and its security extension based on common factor. Frontiers of Computer Science, 16(1), 1-9 (2022). https://doi.org/10.1007/s11704-021-0483-9.
[31] Zuo, C., Shao, J., Wei, G., Xie, M., & Ji, M.: CCA-secure ABE with outsourced decryption for fog computing. Future Generation Computer Systems, 78, 730-738 (2018). https://doi.org/10.1016/j.future.2016.10.028.
[32] Mittelbach, A., & Fischlin, M.: The Theory of Hash Functions and Random Oracles. An Approach to Modern Cryptography, Cham: Springer Nature (2021). https://doi.org/10.1007/978-3-030-63287-8.
[33] Ning, Y., Miao, F., Huang, W., Meng, K., Xiong, Y., & Wang, X.: Constructing ideal secret sharing schemes based on Chinese remainder theorem. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 310-331). Springer, Cham (2018, December). https://doi.org/10.1007/978-3-030-03332-3_12.
[34] Mosca, M., & Verschoor, S. R.: Factoring semi-primes with (quantum) SAT-solvers. Scientific Reports, 12(1), 1-12 (2022). https://doi.org/10.1038/s41598-022-11687-7.
[35] Boudot, F., Gaudry, P., Guillevic, A., Heninger, N., Thomé, E., & Zimmermann, P.: The State of the Art in Integer Factoring and Breaking Public-Key Cryptography. IEEE Security & Privacy, 20(2), 80-86 (2022). https://doi.org/10.1109/MSEC.2022.3141918.
[36] Buchmann, J., Loho, J., & Zayer, J.: An implementation of the general number field sieve. In Annual International Cryptology Conference (pp. 159-165). Springer, Berlin, Heidelberg (1993). https://doi.org/10.1007/3-540-48329-2_14.
[37] Nitaj, A., Ariffin, M. R. B. K., Adenan, N. N. H., Lau, T. S. C., & Chen, J.: Security issues of novel RSA variant. IEEE Access, 10, 53788-53796 (2022). https://doi.org/ 10.1109/ ACCESS. 2022. 3175519.
[38] B. Lynn: the Pairing-Based Cryptography Library. http://crypto.stanford.edu/pbc/ (2013).