








A Secure Data Access Control Scheme Without Bilinear Pairing in Edge Computing

Xiaofei Sheng¹ , Junhua Wu¹ , Guangshun Li^{1,2} , Qingyan Lin¹ ,
and Yonghui Yao¹ 

¹ College of Information Science and Engineering, Qufu Normal University, Rizhao 266700, China
guangshunli@qfnu.edu.cn

² Hong Kong Polytechnic University, Hong Kong, China

Abstract. Edge computing, as an extension of cloud computing, sub-contracts the personal private data to edge nodes on the edge network of Internet of Things (IoT) to decrease transmission delay and network congestion. So, a major security concern in edge computing is access control issues for shared data. In this paper we introduce a scheme without bilinear pairing encryption (Un-BPE) to provide access control in edge and cloud communication. To achieve confidentiality, verifiability and access control, the secret key is generated by Key Trust Authority (KTA), end users and edge node together, and saved in cloud platform; the operations of verification are performed by the adjacent edge node; and the operations of encryption and decryption are performed by the terminal device. We verify the efficiency of our scheme in terms of the security of the encryption algorithm and the performance of the system. The analysis of the proposed scheme reveals better computational efficiency.

Keywords: Access control · Edge computing · Bilinear pairing · Data encryption

1 Introduction

Cloud computing [1] can provide elastic computing resources to users, meet the requirement of the end users. However, the centralized computing systems are starting to suffer from the unbearable transmission latency and degraded service due to the development of IoT and big data. Edge computing is a burgeoning technology with distributed system, which has the characteristics of location awareness, low latency, mobility support, etc. [2]. It can not only process a large

Supported by organization x.

amount of data and improve our quality of life, but also collect real-time data for data monitoring and analysis [3].

Although edge computing network extends computing resources to the edge of the network, greatly improving the resource transmission efficiency [4, 5], but it also has many security issues similar to those in the cloud [6]. Communication security is one of the most important concerns for users when using edge computing to transfer data to the cloud for storage and processing [7]. In the process of communication, edge computing network is threatened by data abnormal change, location privacy disclosure and unauthorized access [8, 9]. Moreover, they are more easily compromised and low-trustworthy since fog nodes are deployed at the network edge with much lower cost than cloud servers [10], they are more vulnerable to attacks and less reliable. At the same time, Alrawais et al. pointed out that the research on the security of fog computing in the IoT is still in its infancy, and faces many security access problems [11].

In order to effectively solve the authentication, privacy protection, access control and other important issues [12] in edge computing. [13] researched the authentication and authorization of IoT, and pointed out that the cost of authentication and authorization should be allocated to the edge nodes. It lays a foundation for researching the authentication and authorization technology in edge computing. [14] proposes a fine-grained data access control protocol based on the properties and cipher text updates, In order to ensure the security of data, outsourcing sensitive data to the cloud storage. But it leads to the transmission delay of the data. Outsourcing most of the encryption and decryption operations to the fog node will lead to the decrypted ciphertext in the process of sending to the user is attacked. Yang et al. proposed a concrete construction with lightweight computational overhead for health IoT system, introducing a semi-trusted computing center to perform most of the heavy calculations in the data encryption phase, which reduces the computational energy consumption of user terminals [15], but it cannot guarantee the correctness of the ciphertext. Chen et al. proposed ABS outsourcing protocols, which greatly reduced the computing overhead of the client side by outsourcing intensive computing to untrusted CSP [16], but the signature of this protocol was of high complexity. Our protocol does not completely rely on CSP for key generation, and delegates some operations to edge nodes and users, which guarantees the security of ciphertext and reduces the computation of edge nodes.

Similar to the cloud servers, edge nodes are not fully trusted as well, data security would raise great concerns from users when they store sensitive data on cloud servers through edge nodes [17, 18]. To ensure secure communication, Al-Riyami and Paterson put forward the certificateless public key cryptography, in which not only ID-based cryptography (IBC) solved the key escrow problem, but also the users private key is generated by the user and the key generation center (KGC) together [19]. This result in KGC cannot obtain the users complete private key, solving the problem of untrusted third parties. Huang et al. proposed a certificateless multi-recipient encryption (AMCLE) protocol that based on bilinear pairing and mapping to point (MTP) hash functions [20], which are

less efficient and time consuming. Li et al. performed a security proof on the certificateless signcryption mechanism proposed by the random oracle model [21], but this protocol requires multiple bilinear mapping operations. Scheme [20, 21] uses bilinear mapping for key generation and signcryption calculation, all of which have low computational efficiency. In addition, existing symmetric [22] ciphers cannot meet the security and privacy requirements of data transmission based on edge computing. Inspired by these questions, we proposed a certificateless multi-receiver scheme without bilinear pairing to edge computing, which uses scalar point multiplication to improve computational efficiency. In addition, in order to conduct the security of the key escrow, the edge node and the end users respectively calculate the private key and the public key, thereby ensuring the security of the key and reducing the users computing. This paper main contributions are as follows: (1) A key authentication scheme without bilinear pairing encryption (Un-BPE) in edge computing is proposed to improve the efficient and security of data access control. (2) Outsourcing part of the encryption and decryption operations to the KTA, reducing the amount of computing by the end user; secret key generation is done by the KTA, the end users and the edge nodes together, ensuring its security.

2 Preliminaries

2.1 Secure Data Access Model

The securely system model consists of core cloud platform(CC), edge node(EN), key trust authority(KTA), IoT data owner(DR) and IoT end user(IE), as shown in Fig. 1.

Core cloud platform (CC): It has high computing power and data storage capacity for storing the final key and ciphertext.

Edge node (EN): It is deployed at the edge of the network and provides various services, includes master edge node EN_m and adjacent edge node EN_a . The edge node is mainly responsible for generating the public key; transmitting the public key and ciphertext through the secure channel.

Key trust authority (KTA): It generates the master key and system parameters and publishes for the system. When the end users request data access, it can generate the partial secret key to ensure the security of sensitive data. We assume that the key trust authority is semi-trusted.

Internet of Things end user (IE): It used to generate the private value and the private value parameter of the end users and store it. After receiving the ciphertext and the secret key from the edge node, To fished the decrypted process and verified it.

Internet of Things data owner (DR): It stores some temporary real-time data in the IoT device to capture resources in the cloud. It used to generate ciphertext and transmit it to the edge node.

2.2 Computational Problems

In what follows, the definitions of computational Diffie-Hellman problem, and assumptions are given.

Definition 1. *Computational Diffie-Hellman (CDH) problem.* Let P be a generator of the additive cyclic group G of order p . Given $P \in G_p, a, b \in Z_p^*$, to meet $Q = aP, R = bP$. Computation of abP is computationally hard by a polynomial time-bounded algorithm. The probability that a polynomial time-bounded algorithm A can solve the CDH problem is defined as:

$$Adv^{CDH}(A) = Pr[A(P, Q, R) = abP | a, b \in Z_p^*, Q, R \in G_p].$$

Definition 2. *Computational Diffie-Hellman assumption.* For any probabilistic polynomial time-bounded algorithm $A, Adv^{CDH}(A)$ is negligible. That is, $Adv^{CDH}(A) \leq \epsilon$. For some negligible function ϵ , every $0 < k^c < 1$, there exists k_0 such that $\epsilon(k) \leq k^c$ for every $k \geq k_0$.

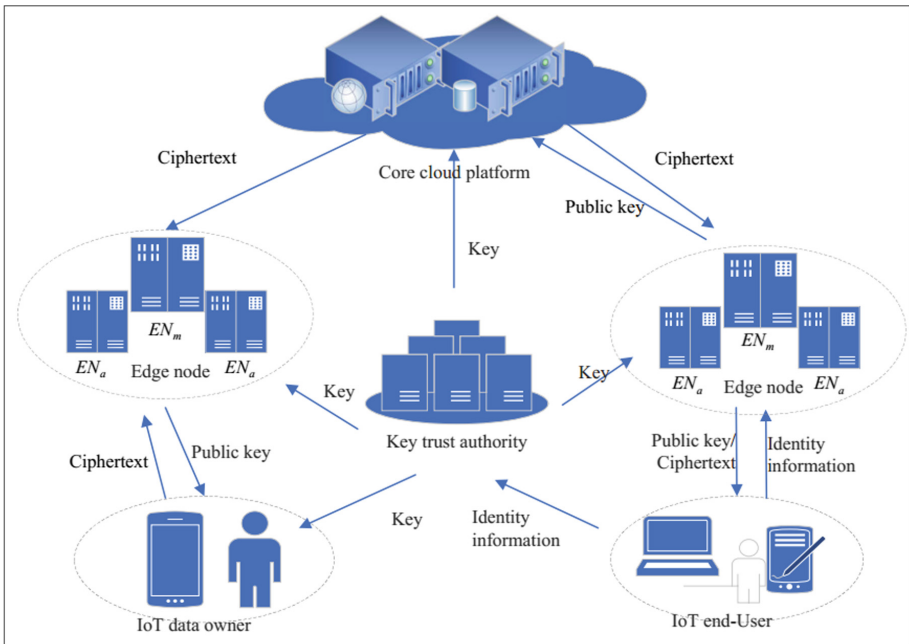


Fig. 1. Secure data access model.

2.3 Security Models

Our scheme can establish secure communication in edge computing network. Therefore, the system should satisfy the following security target.

Message Confidentiality: Users' private data is only provided to legal node. In our proposed system, we use the UN-BPE scheme to ensure the securely communication of the access control.

Message Authentication: Our system should resist the petaggressive, prevents the attacker to change or steal the data information. Therefore, we should introduce a appropriate security mechanisms to ensure the validity of the data. In addition, we have set up message validation steps to enhance key security.

Key escrow security: In our scheme, KTA is adopted for key management and distribution, so as to reduce the risk of user privacy data leakage during data access and ensure the security of data transmission.

3 The Proposed Protocol

In edge computing, reducing complexity is necessary because most devices are resource constrained. Firstly, we propose key authentication without bilinear pairing to guarantee data access control. The end user requests data access, and decrypts the ciphertext with the decryption algorithm. Secondly, the KTA distributes the key to ensure its security; and performs partial encryption operation. Thirdly, the signcryption operation is outsourced to the edge node to improve the computational efficiency. The transmission process of this paper uses the secure channel uniformly. The steps are as follows:

3.1 System Setup

The Setup algorithm is running by KTA and the user to generate the master key, the systems public parameters, users secret value and the secret value parameter.

Algorithm 1. Setup.KTA

- 1: **Input:** k, p, E
 - 2: Selecting the G_p and P
 - 3: Selecting $S \in Z_p^*$
 - 4: Computing $P_{pub} = SP$
 - 5: Selecting $H_0 : \{0, 1\}^* \times Z_p^* \rightarrow Z_p^*, H_1 : \{0, 1\}^* \times Z_p^* \times Z_p^* \rightarrow Z_p^*, H_2 : \{0, 1\}^* \rightarrow Z_p^*, H_3 : \{0, 1\}^* \times G_p \times Z_p^* \times Z_p^* \times \dots \times Z_p^* \rightarrow Z_p^*.$
 - 6: Computing $params = \langle p, F_p, E, G_p, P_{pub}, E_k, D_k, H_0, H_1, H_2, H_3 \rangle$
-

The KTA runs the Algorithm 1, and then saves the master key S secretly, and announces the systems public parameter $Params$. Where k is a generates elliptic curve, p is a prime integer, E is an elliptic curve that defined on finite field F_p , G_p is an additive cyclic group on E and its generator P , the master key S is a randomly chosen integer, P_{pub} is the systems public key, E_k is a

symmetric encryption function, D_k is the corresponding decryption function, H_0, H_1, H_2, H_3 are the secure hash functions.

The user selects a randomly generated integers $v_i \in Z_p^*$, and then edge nodes through a secure channel send $V_i = v_iP$ and the identity information ID_i to KTA, and saves the v_i secretly, where v_i is the secret value of the ID_i , V_i is the secret value parameter of the ID_i .

3.2 Key Generation

The key generation algorithm is running by the user, the edge node and the KTA, and finally generates the users public and private keys.

Algorithm 2. KeyGen.KTA

- 1: Selecting $d_i \in Z_p^*$
 - 2: Computing $Pp_i = H_0(ID_i, V_i)P + d_iP$
 - 3: Computing $Ps_i = H_0(ID_i, V_i)P + (s + d_i)(modp)$
-

After receiving the ID_i and V_i from the user, the KTA runs the Algorithm 2 to obtain the users partial public key Pp_i and the partial private key Ps_i , and transmits Ps_i and Pp_i to the edge node through the secure channel. Where the d_i is a randomly secret integer.

Then the edge node runs the Algorithm 3 to obtain the user public key PK_i and saves it.

Algorithm 3. KeyGen.edge

- 1: checking whether the equation $Ps_iP = Pp_i + P_{pub}$
 - 2: If its really, performing the step 4
 - 3: Else rejecting the Ps_i and Pp_i
 - 4: computing $PK_i = Pp_i + H_1(ID_i, V_i, d_i)V_i$
-

Proof: The establishment of the equation $Ps_iP = Pp_i + P_{pub}$ guarantees the correctness of the partial private key verification of the user. The derivation process can be expressed in Eq. (1).

$$\begin{aligned}
 Ps_i &= (H_0(ID_i, V_i) + s + d_i)P \\
 &= H_0(ID_i, V_i)P + d_iP + sP \\
 &= Pp_i + P_{pub}
 \end{aligned}
 \tag{1}$$

Through the above derivation, the equation $Ps_iP = Pp_i + P_{pub}$ is established. The results show that the key extraction algorithm is correct for the partial private key verification.

Then the user computes $SK_i = H_0(ID_i, PK_i)(PS_i, V_i, d_i)v_i(modp)$ as the private key, and saves it.

3.3 Data Signcryption

The data signcryption algorithm is running by the data owner r , and finally generates the ciphertext and the verification message.

Algorithm 4. Sign-cryption

- 1: Computing $Q_i = PK_i + P_p ub$, where $i = 1, 2, \dots, n$
 - 2: Selecting $w \in Z_p^*$
 - 3: Computing $W = wP$, $F_i = wH_0(ID_i, PK_i)Q_i$ and $\alpha_i = H_1(ID_i, F_i, W)$, where $i = 1, 2, \dots, n$
 - 4: Selecting $\zeta \in Z_p^*$
 - 5: Computing $f(x) = \prod_{i=1}^n (x - a_i) + \zeta(modp) = a_0 + a_1x + \dots + a_{n-1}x^{n-1} + x^n, a_i \in Z_p^*$
 - 6: Computing $k = H_2(\zeta)$, $h = H_3(M || ID_r, \zeta, a_0, a_1, \dots, a_{n-1}, W)$
 - 7: Computing H^{-1} and making ites meet the $hh^{-1} \equiv 1(modp)$, computing $z = h^{-1}(SK_r + w)(modp)$
 - 8: Computing $T = \langle ID_r, W, M, z, j, a_0, a_1, \dots, a_{n-1} \rangle$
 - 9: Computing $B = H_0(ID_r, PK_r,)W$, $J = E_\eta(T)$, $C = (B + J) \oplus P_{pub}$
 - 10: Computing $V = \langle ID_r, C \rangle$
-

The data owner r runs the Algorithm 4, and sends the ciphertext T to adjacent edge node of the end user through cloud platform, and sends the verification message V to the edge node EN_a adjacent to the master edge node EN_m . Where M is the plaintext.

3.4 Data Verification

The data verification algorithm is running by the adjacent edge node EN_a , and finally computes the T' .

Algorithm 5. Verification

- 1: Computing $B = H_0(ID_r, PK_r,)W$,
 - 2: Computing $J' = C \oplus P_{pub} - B$,
 - 3: Computing $T' = D_\eta(J')$
-

The adjacent edge node EN_a runs the Algorithm 5, and sends T' to the adjacent edge node of end user through the cloud platform. The adjacent edge nodes verify the $T' = T$ is true. If not, output terminator “ \perp ”. Else, the edge node sends the ciphertext T to end user.

3.5 Data Decryption

The data decryption Algorithm 6 is running by the end user to obtain the plain-text M to decryption.

Algorithm 6. De-cryption

- 1: Computing $F_i = SK_iW$, $\alpha_i = H_1(ID_i, F_i, W)$
 - 2: Computing $f(x) = \prod_{i=1}^n (x - \alpha_i) + \zeta(modp) = a_0 + a_1x + \dots + a_{n-1}x^{n-1} + x^n$ and $\zeta = f(\alpha_i)$
 - 3: Computing $k = H_2(\zeta)$
 - 4: Computing $h' = H_3(M||ID_r, \zeta, a_0, a_1, \dots, a_{n-1}, W)$
 - 5: Checking the $h' = h$ holds
 - 6: If its really, performing the next step
 - 7: Else rejecting the M and exit the process
 - 8: Obtainable the PK_r and checking $hzP = H_0(ID_r, PK_r)(PK_r + P_{pub}) + W$ holds
 - 9: If its really, then received the plaintext M and exit the process
 - 10: Else rejecting the M and exit the process
-

Where PK_r is the public key of the data user.

Proof: The correctness of the decryption algorithm is guaranteed by $h' = h$ and $hzP = H_0(ID_r, PK_r)(PK_r + P_{pub}) + W$, and deductions that these two equations hold are shown in the following.

For every receiver r_i , with the ciphertext T , it has $F_i = SK_iW$ and $\alpha_i = H_1(ID_i, F_i, W)$. Then with the α_i , it can compute $\zeta = f(\alpha_i)$, and then obtain $k = H_2(\zeta)$. Finally it has $h' = H_3(M||ID_r, \zeta, a_0, a_1, \dots, a_{n-1}, W)$. So the equation $h' = h$ holds.

When decrypting the identity ID_r of the data owner, the end user can obtain the data owners public key and has computing the Eq. (2).

$$\begin{aligned}
 hzP &= H_0(ID_r, PK_r)(PK_r + P_{pub}) + W \\
 &= SK_rP + W \\
 &= H_0(ID_r, PK_r)(Ps_r, +H_1(ID_r, V_r, d_j)v_r)P + W \\
 &= H_0(ID_r, PK_r)(Pp_r, +H_1(ID_r, V_r, d_j)V_r + P_{pub}) + W \\
 &= H_0(ID_r, PK_r)(PK_r + P_{pub}) + W
 \end{aligned}
 \tag{2}$$

4 Algorithms Analysis

4.1 System Cryptanalysis

In this section, we analyze the safety strength of our proposed scheme from the aspects of message confidentiality, authentication and unforgeability.

Message Confidentiality. In our scheme, we use the Un-BPE to ensure the safety of the key. Our model provides a model for encryption and decryption data, and Un-BPE provides the key of encryption and decryption. Since the public key is produced jointly by the KTA and the edge nodes, and the secret key is produced jointly by the KTA and the end user and kept privately by the end user. Thus spite nodes cannot gain the public key.

Authentication. Suppose that the cloud could send the key to the DR through the edge node, the DR use the algorithm Sign-cryption to encrypt data, and then disclosure the message. When the edge nodes gain the message, they need the public parameters, which are computed by algorithm Setup.KTA. At the same time, the edge nodes gain the verification message, and then it verify the signature via Sign-cryption. If passed, the edge node send the ciphertext and the end users decrypt the ciphertext to obtain the plaintext M; otherwise, it is termination.

Unforgeability. A malicious node must have the user's private key to product an available signature for a legitimate user, but the malicious node cannot deduce the private key. A malicious node cannot create a new available ciphertext from another user's ciphertext. If the malicious nodes alter the ciphertext, the receiver can use the algorithm to examine that the ciphertext is vicious. Therefore, we said that our proposed scheme is unfalsifiable.

4.2 Efficiency Comparison

Through the simulation experiment, some basic operation consumption time is tested: T_p refers to the time consumed by the table bilinear pair operation; T_e refers to the time consumed by the table modulus power operation; T_{pe} refers to the table bilinear pair exponential operation time; T_a refers to the time consumed by the point addition operation; T_{sm} refers to the table scalar point multiplication operation Time consuming; T_h refers to the time it takes for the hash function to map to a point (MTP). The experimental environment is Dell notebook (i5-4200U CPU@1.60 GHz 8 GB memory Windows 7 operating system), the time spent on the above basic operations is shown in Table 1. It is worth noting that we only consider the time of these operations as defined in Table 1, and don't consider the time of other operations, because their run time is negligible compared to the operations defined in Table 1.

Table 2 shows the efficiency comparison between our protocol and the encryption and decryption phases of the [20, 24–26] protocol. Where n is the number of data recipients. Since the system setup and key generation phases are primarily performed on the KTA, and the safety of these two phases is more important, which was discussed in Chapter 5, we study the efficiency of the encryption and decryption phases. The protocols of [24] and [20] are based on certificateless encryption with bilinear pairing encryption algorithm, our protocol, [25] and [26] are based on the scalar point multiplication in ECC for encryption and

Table 1. Time consumed by basic operations/ms

Operation	Running time
T_p	12.019
T_e	34.068
T_{pe}	9.045
T_a	0.023
T_{sm}	6.032
T_h	6.720

decryption operations. As can be seen from Table 2, our protocol efficiency is more effective than [20, 24], but lower than [25, 26]. The reason is that our program decryption process has the steps to verify the source of the message, but the program [25, 26] does not.

Table 2. Comparison of algorithm efficiency

Protocol	Encryption	Decryption
[24]	$2(n+1)T_e + (n+1)T_{sm}/2$	$T_{sm}/2 + T_{pe} + 2T_p + T_a$
[20]	$(n+1)T_{sm}/2 + nT_{pe} + nT_p + nT_h$	$T_{sm}/2 + T_p$
[25]	$(n+1)T_{sm}/2 + 2nT_a$	$T_{sm}/2$
[26]	$(3n+1)T_{sm}/2 + nT_a$	T_{sm}
Our protocol	$(n+1)T_{sm}/2 + nT_a$	$T_{sm} + T_a$

When the number of our data receivers is $n=1$, the comparison between our protocol and [20, 24–26] protocol is shown in Fig. 2. It can be clearly seen from Fig. 2 that our protocol is based on double-based without certificate. Linear encryption protocols are more efficient in the encryption and decryption phases. achieve the anonymity of the data receiver (end user), which means that no other device except the sender knows the identity of receiver. The protocol [24, 25] doesn't consider the recipients anonymity, which means that the recipients identity in his ciphertext is directly revealed. Our protocol and [25] have partial private key verifiability, which can effectively prevent the KTA from generating false partial private keys to spoof end users. However, since protocols [20, 24] and [26] have no partial private key verifiability, they cannot prevent malicious attacks. Our protocol and [24] implement the signature function to ensure the reliability of the message and prevent the attacker from pretending to send the message as the sender. However, the protocol [20, 25, 26] does not have this function. In short, our solution has more features, is safer and more suitable for practical applications.

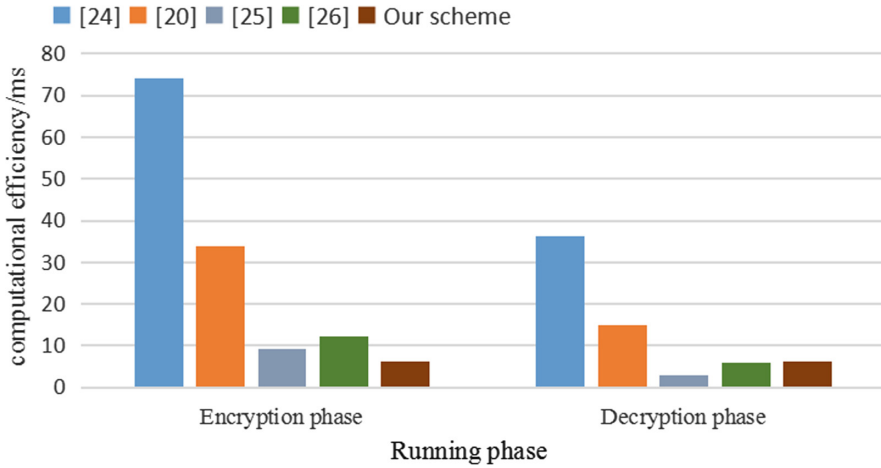


Fig. 2. Efficiency comparison.

5 Conclusion

In this paper, we propose to apply a non-bilinear pairing encryption protocol to edge computing. On the one hand, part of the calculation and storage is outsourced to the central edge node and key trust authority, so as to reduce the computational overhead of the end user, abandon the traditional bilinear pairing algorithm to improve efficiency; on the other hand, based on the Un-BPE algorithm, reduce The number of traditional ECC-encrypted scalar points, using KTA and the user to calculate the key at the same time, enhances the security of data access in edge computing.

References

1. Li, J., Yao, W., et al.: Flexible and fine-grained attribute-based data storage in cloud computing. *IEEE Trans. Serv. Comput.* **10**(5), 785–796 (2017)
2. Zhu, J.T., Shi, T., et al.: Task scheduling in deadline-aware mobile edge computing systems. *IEEE Internet Things J.* **6**(3), 4854–4866 (2019)
3. Rathore, M., Paul, A., et al.: IoT-based big data: from smart city towards next generation super city planning. *Int. J. Seman. Web Inf. Syst.* **13**(1), 28–47 (2017)
4. Li, G., Xi, S., et al.: Resource scheduling based on improved spectral clustering algorithm in edge computing. *Sci. Program.* **5**, 1–13 (2018)
5. Li, G., Liu, Y., et al.: Methods of resource scheduling based on optimized fuzzy clustering in fog computing. *Sensors* **19**(9), 2122 (2019)
6. Xi, Y., Qi, Y., et al.: IoT-based big data: from smart city towards next generation super city planning. *Complexity*, 1–9 (2017)
7. Shiraz, S.: The extended cloud: review and analysis of mobile edge computing and fog from a security and resilience perspective. *IEEE J. Sel. Areas Commun.* **35**(11), 2586–2595 (2017)

8. Zhang, J., Chen, B., et al.: Data security and privacy-preserving in edge computing paradigm: survey and open issues. *IEEE Access* **6**, 18209–18237 (2018)
9. Cai, Z., Zheng, X., et al.: A private and efficient mechanism for data uploading in smart cyber-physical systems. *IEEE Trans. Netw. Sci. Eng.*, 1 (2018)
10. Hoe, S.L.: Defining a smart nation: the case of Singapore. *J. Inf. Commun. Ethics Soc.* **14**(4), 323–333 (2016)
11. Alrawais, A., Alhothaily, A., et al.: Fog computing for the internet of things: security and privacy issues. *IEEE Internet Comput.* **21**(2), 34–42 (2017)
12. Gai, K., Qiu, M., et al.: Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers. *IEEE Trans. Ind. Inf.* **14**(8), 3590–3598 (2018)
13. Kim, H., Lee, E., et al.: Authentication and authorization for the internet of things. *IT Prof.* **19**(5), 27–33 (2017)
14. Huang, Q., Yang, Y., et al.: Secure data access control with ciphertext update and computation outsourcing in fog computing for internet of things. *IEEE Access* **5**, 12941–12950 (2017)
15. Yang, Y., Zheng, X., et al.: Lightweight distributed secure data management system for health internet of things. *J. Netw. Comput. Appl.* **89**, 26–37 (2016)
16. Chen, X., Li, J., et al.: Secure outsourced attribute-based signatures. *IEEE Trans. Parallel Distrib. Syst.* **25**(12), 3285–3294 (2014)
17. Lee, K., Kim, D., et al.: On security and privacy issues of fog computing supported Internet of Things environment. In: *International Conference on the Network of the Future*, Montreal, pp. 1–3 (2015)
18. Roman, R., Lopez, J., et al.: Mobile edge computing, Fog et al.: a survey and analysis of security threats and challenges. *Fut. Gener. Comput. Syst.* **78**(TP.2), 680–698 (2018)
19. Alriyami, S.S., Paterson, K.G., et al.: Certificateless public key cryptography. *Asiacrypt* **2894**(2), 452–473 (2003)
20. Huang, Y.H., Huang, S.S., et al.: Efficient anonymous multireceiver certificateless encryption. *IEEE Syst. J.* **11**(4), 2602–2613 (2017)
21. Li, F., Shirase, M., et al.: Certificateless hybrid signcryption. *Math. Comput. Model.* **57**(3), 324–343 (2013)
22. Lester, C.G., Nachman, B., et al.: Bisection-based asymmetric MT2 computation: a higher precision calculator than existing symmetric methods. *J. High Energy Phys.* **3**, 1–16 (2015)
23. Yuan, Y., Li, D., Tian, L., Zhu, H.: Certificateless signature scheme without random oracles. In: Park, J.H., Chen, H.-H., Atiquzzaman, M., Lee, C., Kim, T., Yeo, S.-S. (eds.) *ISA 2009*. LNCS, vol. 5576, pp. 31–40. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02617-1_4
24. Selvi, S.S.D., Vivek, S.S., Shukla, D., Rangan Chandrasekaran, P.: Efficient and provably secure certificateless multi-receiver signcryption. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) *ProvSec 2008*. LNCS, vol. 5324, pp. 52–67. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88733-1_4
25. Islam, S.H., Khan, M.K., et al.: Anonymous and provably secure certificateless multireceiver encryption without bilinear pairing. *Secur. Commun. Netw.* **8**(13), 2214–2231 (2015)
26. He, D., Wang, H., Wang, L., Shen, J., Yang, X.: Efficient certificateless anonymous multi-receiver encryption scheme for mobile devices. *Soft Comput.* **21**(22), 6801–6810 (2016). <https://doi.org/10.1007/s00500-016-2231-x>