Hindawi
Shock and Vibration
Volume 2021, Article ID 3920579, 11 pages
https://doi.org/10.1155/2021/3920579

Research Article

The New Method of Sensor Data Privacy Protection for IoT

Yue Wu,[1,2] **Liangtu Song**,[1,2] **and Lei Liu**[1,2]

[1]*Institute of Intelligent Machines, and Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, Anhui, China*
[2]*University of Science and Technology of China, Hefei 230026, Anhui, China*

Correspondence should be addressed to Yue Wu; wyw533k@mail.ustc.edu.cn

Received 24 May 2021; Revised 15 June 2021; Accepted 21 June 2021; Published 22 July 2021

Academic Editor: Chaoqun Duan

This article introduces the new method of sensor data privacy protection method for IoT. Asymmetric encryption is used to verify the identity of the gateway by the sensor. The IoT gateway node verifies the integrity and source of the data, then creates a block, and submits the block chain transaction. In order to avoid tracking the source of the data, a ring signature is used to anonymize the gateway transaction. The proxy re-encryption method realizes the sharing of encrypted data. On the basis of smart contracts, attribute-based data access control allows decentralized applications to finely control data access. Through experiments, the effects of sensor/gateway verification, transaction signatures, and sensor data encryption on performance are discussed. The results show that transaction delays are all controlled within a reasonable range. The system performance achieved by this method is also relatively stable.

1. Introduction

Various sensors have penetrated into all aspects of our lives, and many of them are continuously collecting our information [1]. In this context, privacy issues related to the Internet of Things, especially consumer Internet of Things, have become the focus of attention from all walks of life [2]. As a kind of microcomputer terminal, the Internet of Things device exists independently of the computer network but is closely connected with the Internet [3]. Today, IoT sensors have penetrated into various industries, from industrial automation to medical equipment and then to the financial industry [4]. In short, there will be sensors wherever humans live [5].

Privacy protection is an important issue in IoT systems but has different references in different scenarios. For example, in the smart grid, the leakage of household energy consumption data and other information may cause hidden dangers to the safety of household personnel and property [6–8]. In smart medical care, the leakage of health data generated by patients' wearable devices causes personal safety and ethical issues. Thus, following the above two cases, the privacy protection issue is considered related to the interests of the owner of the sensor data. This kind of privacy protection is called active privacy protection wherein the owners of sensor data have the opportunity to take necessary measures to protect their privacy [9, 10]. However, in public video surveillance applications, the privacy problem caused by face recognition is another situation. In this scenario, the issue of privacy protection is not an issue related to the interests of the owner of the sensor data (that is, the operator of the monitoring system) [11]. Whether to protect the privacy of the monitored person depends on the law and the level of the operator. Similarly, electronic license plate applications exist [12]. This kind of privacy protection is called passive privacy protection wherein the object whose privacy is compromised is powerless to solve this problem. For these two privacy protection problems, the problems to be solved and the directions to be considered are different. Thus, obvious differences are found in the solutions. For example, automatic coding method used in public video surveillance is rare in active privacy protection [13–15].

This article introduces how to solve the privacy protection about IoT sensor data based on blockchain [16]. The ring signature realizes the anonymization of gateway transactions, prevents data sources from being tracked, and

solves the anonymization problem of blockchain-based IoT users [17]. Through asymmetric encryption, the sensor verifies the identity of the gateway and encrypts the sensor data [18]. The gateway can create a block and submit a blockchain transaction after verifying the integrity and source of the data. Finally, combined with data access control and data encryption sharing, decentralized applications perform fine-grained control over data access.

2. Method Architecture

The method architecture is shown in Figure 1. The system consists of sensors, gateways, blockchain, and various decentralized applications. First of all, the sensor and the gateway need to perform bilateral authentication, which can not only prevent the sensor from accessing the fake gateway, but the gateway also filters out offensive sensor data. After the identity verification is passed, Hash-based message authentication code is used to verify the source and integrity of the data collected by the sensor. The sensor uploads the data to the gateway. In order to protect the privacy of the sensor data, the gateway will first encrypt the data with a public key, use a ring signature to verify the anonymity of the transaction, and finally submit it to the blockchain. Based on smart contracts, an attribute-based control method is adopted. The encrypted data on the blockchain are decrypted and used by decentralized applications by proxy re-encryption.

2.1. Sensor-Gateway Authentication. To avoid the leakage of private data caused by the sensor's access to the wrong gateway, this study adopts the Elliptic Curve Diffie–Hellman (ECDH) protocol combined with an asymmetric encryption method to realise the sensor's authentication of the gateway's identity and negotiate a shared key in the process.

ECDH is a variant of the Diffie–Hellman (DH) protocol that uses elliptic curve cryptography. The difference between the two is that ECDH is based on the elliptic curve discrete logarithm problem, whereas the DH protocol is based on the discrete logarithm problem. Similar to the DH protocol, the two parties in ECDH communication use their elliptic curve key pairs to negotiate a shared key on an insecure channel. This key can be used for the symmetric encryption of subsequent communications between the two parties. The negotiation process is shown in Figure 2.

The sensor and the gateway share a set of elliptic curve domain parameters (p, a, b, G, n, h). The sensor generates a random number $d_{a,g}$, $(d_g \in [2, n-1])$, then the sensor sends Q_s to the gateway, and the gateway sends Q_g to the sensor. Finally, both parties obtain the same shared key (i.e., $K = K_s = K_g$).

The ECDH-based IoT device authentication to the gateway and the key negotiation process is shown in Figure 3.

(i) The sensor side generates a random number d_s, encrypts Q_s with the public key of the gateway, and sends it

(ii) The gateway side generates a random number, d_g, and sends back $Q_s \| Q_g$ in plaintext

(iii) If the sensor successfully parses out Q_s, the gateway identity is correct

The above process shows that, when the system is initialised, an ECC key pair needs to be allocated to the gateway, and the sensor needs to know the public key that it needs to access the gateway.

2.2. Sensor Data Encryption. The gateway creates and submits a blockchain transaction after verifying the integrity and source of the sensor data. To protect the privacy of sensor data when constructing a transaction, this article first uses the ECC public key to encrypt sensor data. The principle of public key encryption is as follows:

(i) The 32-bit input data m are converted into point M on the elliptic curve, where f represents the selected elliptic curve equation:

$$M = (m, f(m)). \tag{1}$$

(ii) A random number $r \in [2, n-1]$ is taken, where n represents the order of the selected elliptic curve equation.

(iii) The first part $C1$ of the encrypted output is calculated, where G is the Genrator of the elliptic curve:

$$C1 = r \times G. \tag{2}$$

(iv) The second part $C2$ of the encrypted output is calculated, where K represents the public key used for encryption:

$$C2 = M + r \times K. \tag{3}$$

(v) The encrypted outputs $C1$ and $C2$ are obtained.

The process of decrypting with the ECC private key is as follows:

(i) M is calculated according to the following formula, where k is the private key used for decryption:

$$M - C2 - k \times C1. \tag{4}$$

(ii) The original information m is extracted from M. For example, M is taken to obtain the x coordinate. The above process is only applicable to the encryption and decryption processing of the input message m with a length of 32 bytes. The message m of any length should be divided if the 32-byte fragments $m_1, \ldots, m_n$ are viewed. The encryption process is performed on each fragment at a time, and the encryption result is obtained.

$$C = [C1, C2_1, \ldots, C2_n]. \tag{5}$$

When decrypting, $C1$ is used to process $C2_1, \ldots, C2_n$ sequentially and obtain the result.

2.3. Anonymization of Transaction Sources. To avoid tracking and identifying the source of the data (gateway), the Borromean ring signature method is used to anonymize gateway transactions. Ring signature is a special group signature method. The difference is that a ring signature does not require an additional group manager. With ring signature, the real signer can be hidden behind a set of public keys (address), thus realising the transaction anonymity of the true initiator.

Assuming that the message to be signed is m, the signer's private key is S_s, and the selected ring members are $P_1, P_2, \ldots, P_r$, the calculation process of the ring signature is as follows:

(i) The hash of the message m is calculated as the symmetric key k as follows:

$$k = h(m). \tag{6}$$

(ii) The signer chooses a random value $v \in [0, 1]$.

(iii) The signer chooses a random value for other members (i.e., $x_i \in [0, 1], 1 \le i \le r, i \ne s$).

(iv) y_s is solved, which makes the following formula true:

$$C_{k,v}(y_1, y_2, \ldots, y_k) = v,$$
$$C_{k,v}(y_1, y_2, \ldots, y_k) = E_k(y_r \oplus E_k(y_{r-1} \oplus E_k(\ldots \oplus E_k(y_1 \oplus v)))). \tag{7}$$

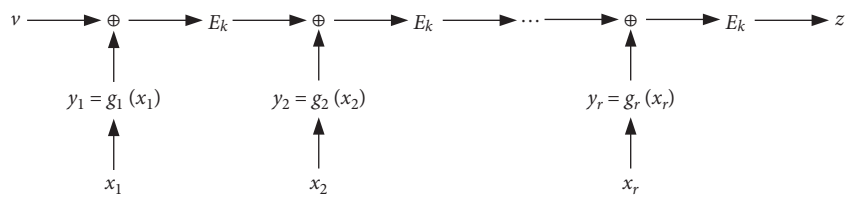
(v) Signer trapdoor permutation and inversion are used:

$$x_s = g_x^{-1}(y_s). \tag{8}$$

(vi) The ring signature is the output:

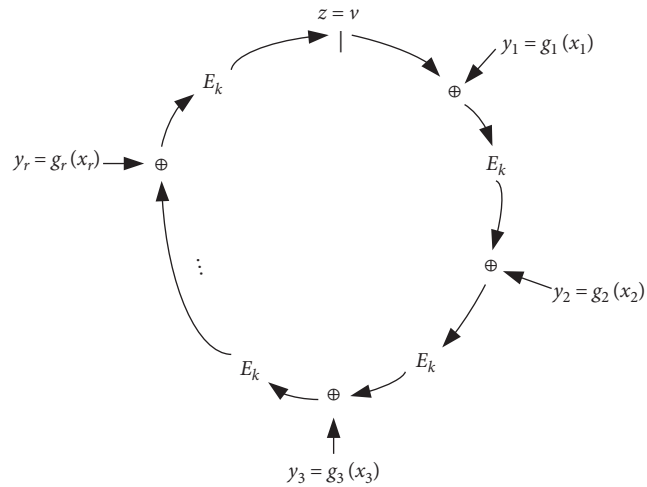
$$(P_1, P_2, \ldots, P_r; v; x_1, x_2, \ldots, x_r). \tag{9}$$

The equation calculated in step 4 can be shown in the figure below, where E_k is the symmetric encryption function:



(10)

The verification process of the ring signature is as follows:



(11)

The equation can be expressed as a ring as follows:

(i) Calculate y_i as follows:

$$y_i = g_i(x_i). \tag{12}$$

(ii) Calculate the encryption key k as follows:

$$k = h(m). \tag{13}$$

(iii) Verify the ring equation as follows:

$$C_{k,v}(y_1, y_2, \ldots, y_k) = v. \tag{14}$$

2.4. Sensor Data Shared. The encrypted sensor data can be decrypted and used by the encryptor. Third-party use must be considered in many cases. However, directly sharing the private key of the encryptor is not safe. Thus, this article uses a proxy re-encryption method to realise the sharing of encrypted data. The proxy re-encryption process is shown in Figure 4. In this article, the blockchain node acts as a re-encryption agent.

Users A and B hold key pairs (sK_A, pK_A) and (sK_B, pK_B), respectively. User A uses his public key pK_A to encrypt the inscription data m to obtain the ciphertext C_A on the chain. When user A needs to share his data with user B, user A generates a re-encryption key $rK_{A \longrightarrow B}$ for user B and provides it to the blockchain node to re-encrypt the specified ciphertext to C_B. After receiving it, user B uses his private key. The key sK_B can be decrypted.

(i) b: user B's private key

(ii) a: user A's private key

(iii) q: order

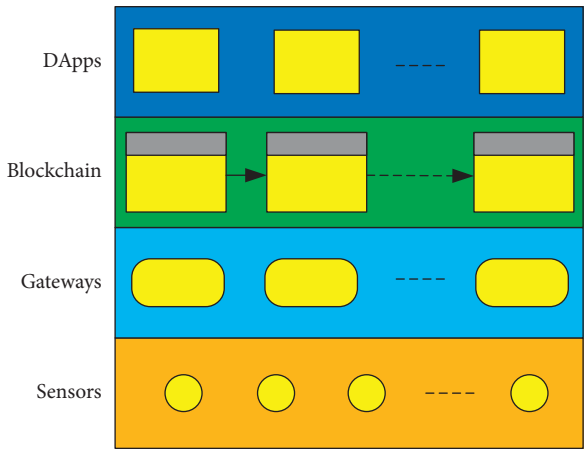


Figure 1: Method architecture.

Given that two sets of private keys are required to generate the re-encryption key and to avoid using B's private key, user A needs to generate an additional temporary private key and provide it to user B after encrypting it with B's public key.

To access the data of the specified sensor, decentralized applications need to request sensor data access permission from the gateway to which the sensor belongs, that is, to send the following message to the specified gateway. The fields of the message are shown in Table 1.

If the gateway agrees for decentralized applications to access the specified sensor, the gateway submits an authorisation transaction to the PAP contract on the chain, granting the requester the access permission to the specified sensor. The transaction parameters are as follows:

(i) Target contract: PAP
(ii) Contract action: grant
(iii) Action parameters:
(iv) Sensor: authorised sensor ID
(v) User: requester ID
(vi) Rk: re-encryption key generated for the requester
(vii) Sk: decryption key generated by the requester, which is encrypted with the requester's public key

After the transaction is confirmed on the chain, the pap contract is triggered to modify the access strategy of the specified sensor. The process is shown in Figure 5.

After the above transaction is confirmed, the requesting party can encrypt and decrypt the specified sensor data to obtain plaintext data. Decentralized applications request the latest sensor data by sending the following message to the blockchain node. The fields of the message are shown in Table 2.

After the node receives the above request, it will first check whether the requester has the permission to access the requested sensor. If the permission is granted, it will return the latest data (encrypted form) of the sensor and the key authorised by the gateway to the requester. The fields of the message are shown in Table 3.

The sequence diagram of the above process is shown in Figure 6.

2.5. Data Access Control. ABAC can finely control access to resources and mainly includes four components, namely, policy enforcement point (PEP), policy decision point (PDP), policy administration point (PAP), and policy information point (PIP). The data access process is shown in Figure 7.

(i) PEP is responsible for receiving user requests, invoking PDP permission evaluation and determining whether to allow access to specified resources based on PDP evaluation results
(ii) PDP evaluates the access request based on the rule base and returns the evaluation result (i.e., denying or allowing access)
(iii) PAP is the management interface of rules provided for administrators, such as adding new access policies and updating designated access policies
(iv) PIP provides out-of-core attribute information for PDP

3. Experimental Evaluation

3.1. Verification of Encrypted Data Utilisation Process. The experimental configuration is as follows:

(i) Synthesis of sensor data: open
(ii) Verification of sensors and gateway devices: open
(iii) Encryption of transactions: open
(iv) Data encryption method: ECC

The data chaining process in the experiment is described as follows:

(i) The sensor verifies the identity of the gateway and negotiates a shared key
(ii) The sensor submits data to the IoT gateway
(iii) The IoT gateway node verifies the integrity and source of the data and rejects the data if the verification fails
(iv) The gateway encrypts sensor data, generates new transactions, and performs ring signatures
(v) After the blockchain node verifies that the transaction is correct, the node will queue the transaction up in the buffer pool
(vi) Blockchain nodes generate blocks to confirm transactions

The access authorisation process in the experiment is as follows:

(i) The gateway submits an authorisation transaction to the blockchain node and grants the data access rights of 80000# sensor to decentralized applications.

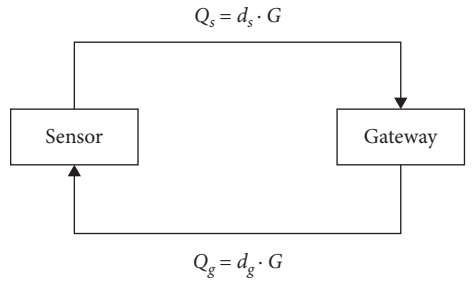


Figure 2: Key agreement.

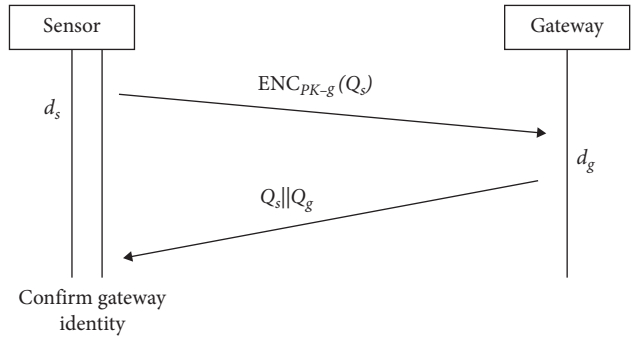


Figure 3: IoT device authentication to the gateway and key agreement.

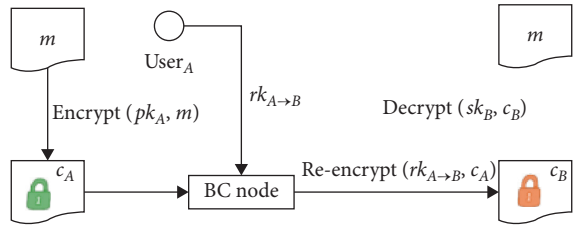


Figure 4: Proxy re-encryption.

(ii) Decentralized applications regularly submit query requests to blockchain nodes and obtain encrypted sensor data and the proxy re-encryption group.

(iii) Decentralized applications perform re-encryption first, decrypts the data, and displays the decrypted plaintext. The experimental results are in line with expectations. The screenshot is shown in Figure 8.

3.2. The Impact of Encryption on Performance. The program is implemented in Python language, and the experiment is mainly carried out in the following aspects: the impact of transaction signatures on performance and the impact of sensor data encryption on performance.

The system has set up 50 sensor nodes, 10 gateways, and 4 blockchain nodes. The system can set parameters in advance, set the block generation cycle of the blockchain node to 5 s, set the sensor report data cycle to 1 s, enable sensor data integration, configure whether to verify the sensor/gateway, and configure transaction signatures and sensor encryption methods.

3.2.1. The Impact of Transaction Signatures on Performance. We use different signature methods to discuss the impact on system performance. The parameters are as follows:

(i) Cycle of block generation: 5 s

(ii) Cycle of sensor data submission: 1 s

(iii) Synthesis of sensor data: open

(iv) Verification of sensors and gateway devices: close

(v) Encryption of transactions: close

(vi) Data encryption method: none/ECC/ring

The transaction delay time statistics of the no-signature method and the ECC signature method are shown in Figures 9–12

Figures 9–12 show the influence of ECC signature on transaction delay time. When no signature is added, all transactions are confirmed within 5 s of the block generation period. When the ECC signature is added, the maximum transaction confirmation time is delayed to more than 7 s. That is, given that ECC signature requires a certain processing time, the simulation system has been overloaded

Table 1: The format of the message sent by decentralized applications.

#	Field	Type	Explanation
1	Type	String	Message type and value: sensor_rights_request
2	Payload	Bytes	The message payload is as follows: (i) Requestor: requester ID (ii) Requestor_pk: requestor public key (iii) Sensor: target sensor number

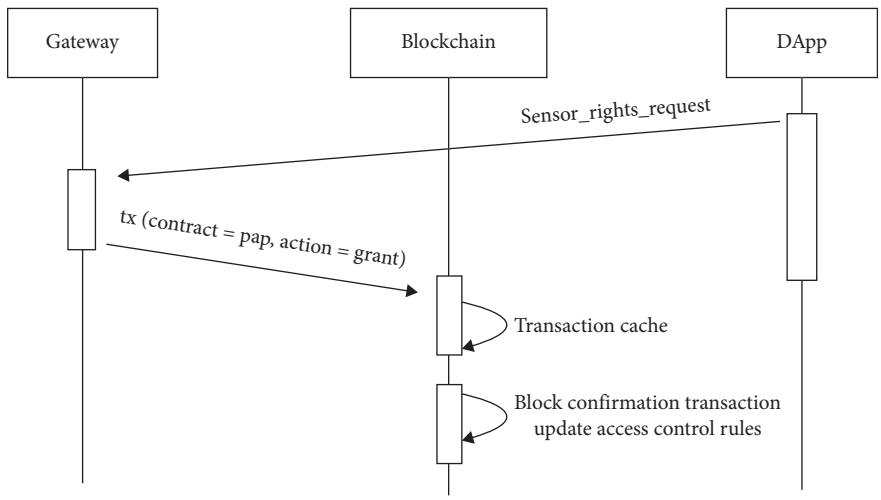


Figure 5: Transaction process.

Table 2: The format of the request sensor data sent by the decentralized applications.

#	Field	Type	Explanation
1	Type	String	Message type and value: state_request
2	Payload	Bytes	The message payload is as follows: (i) Target contract: SCADA (ii) Contract view: sensor_latest (a) View parameters: (b) Requestor: requestor ID (c) Sensor: requested sensor ID

Table 3: Node returns sensor data.

#	Field	Type	Explanation
1	Type	String	Message type and value: state_response
2	Payload	Bytes	The message payload is as follows: (i) Encrypted sensor data (ii) Data re-encryption key (iii) Data decryption key (encrypted by the requester's public key)

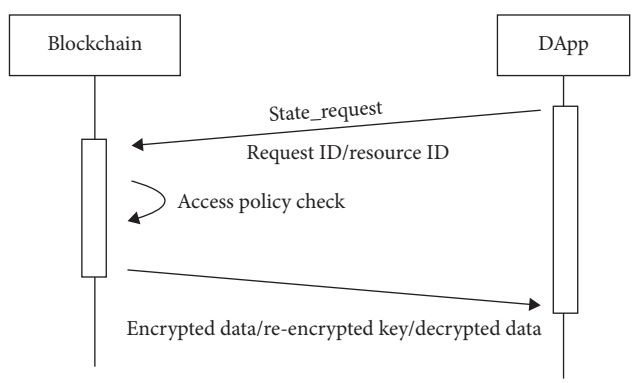


Figure 6: Access policy check.

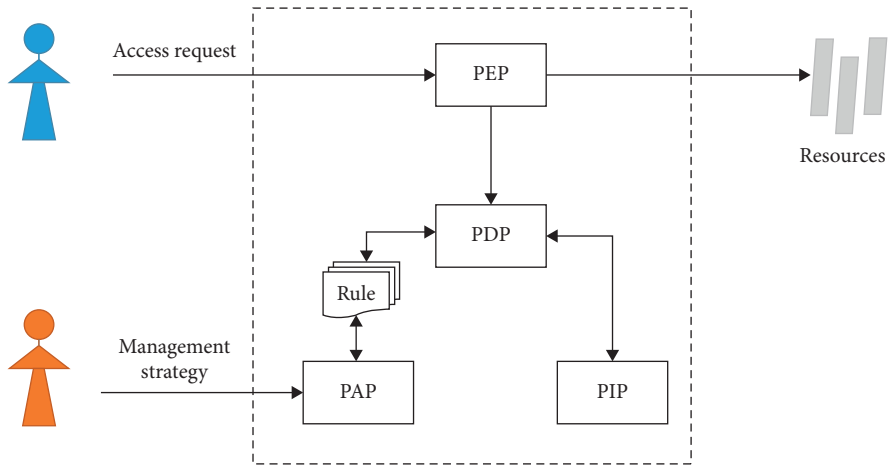


Figure 7: Data access control.

Figure 8: System operation result.

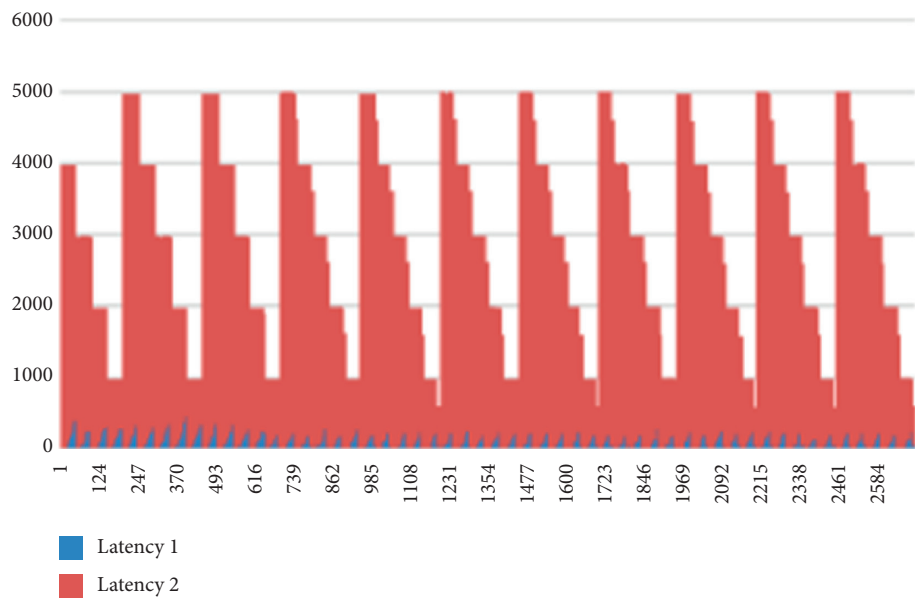


Figure 9: Transaction delay (no signature, $T = 5$ s).

when the block generation period is set to 5 s. When the ring signature method is used, this overload phenomenon is more obvious because the ring signature requires more processing time. Figures 13 and 14 show that the maximum transaction delay time has exceeded 20 s.

The block generation period is set to 30s, and the experiment of the ring signature method is re-run. The results are shown in Figures 15 and 16

The transaction delays are kept within the block generation period, and the system is operating normally.

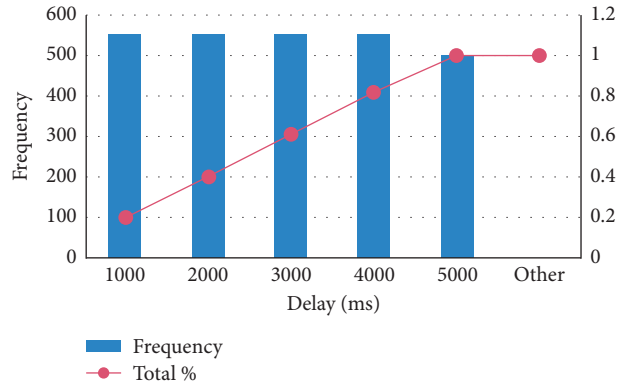


Figure 10: Transaction delay histogram (no signature, $T = 5$ s).

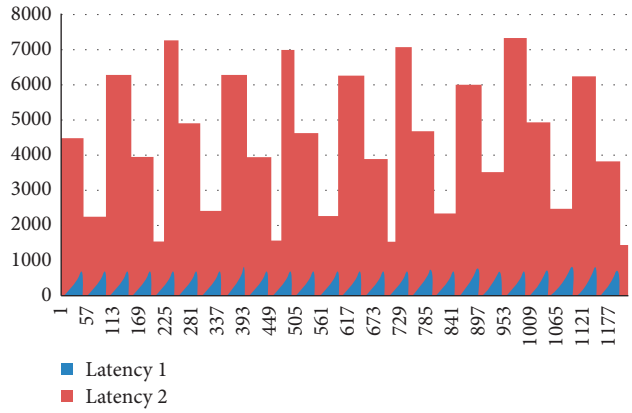


Figure 11: Transaction delay (ECC signature, $T = 5$ s).

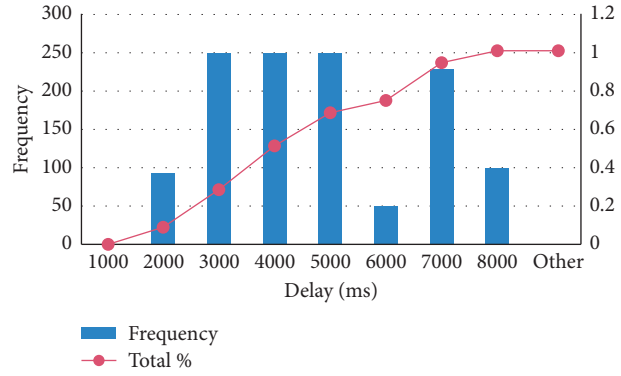


Figure 12: Transaction delay histogram (ECC signature, $T = 5$ s).

3.2.2. The Impact of Sensor Data Encryption on Performance. Sensor data encryption and nonencryption methods are used to investigate the impact of this link on performance. The experimental parameters are as follows:

(i) Cycle of block generation: 5 s

(ii) Cycle of sensor data submission: 1 s

(iii) Synthesis of sensor data: open

(iv) Verification of sensors and gateway devices: close

(v) Encryption of transactions: close

(vi) Data encryption method: none/ECC/ring

The experimental results are shown in Figures 17 and 18.

Figures 17 and 18 show that the encryption of sensor data has little effect on transaction delay. Figure 17 shows the corresponding transaction delay histogram statistics and transaction delay cumulative ratio statistics when sensor data encryption is not enabled. Figure 18 shows the statistics of the transaction smoking, eating, and releasing graphs and

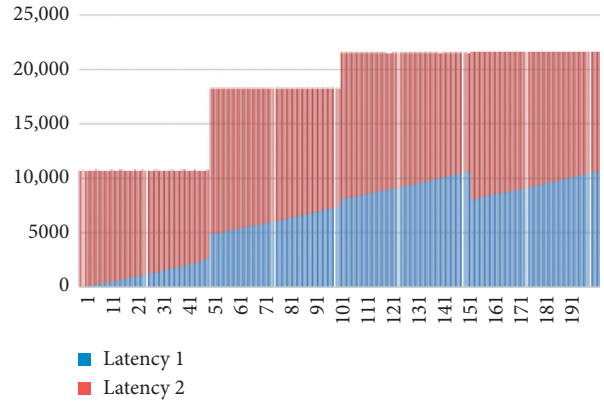


Figure 13: Transaction delay (ring signature, $T = 5$ s).

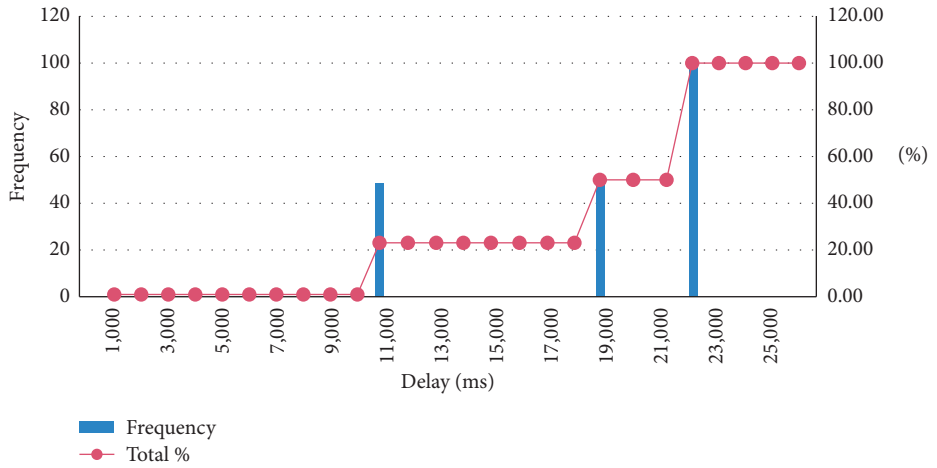


Figure 14: Transaction delay histogram (ring signature, $T = 5$ s).

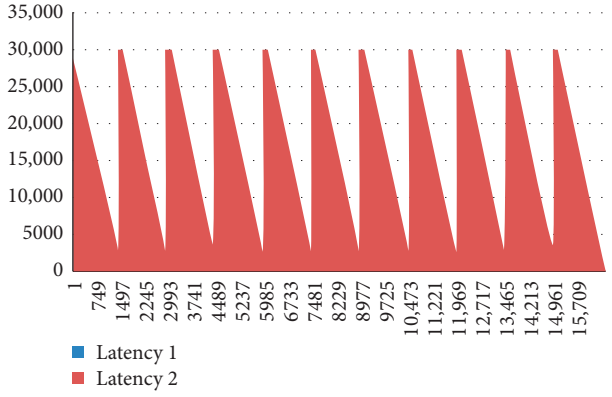


Figure 15: Transaction delay (ring signature, $T = 30$ s).

the cumulative ratio of transaction delays after the sensor data encryption is turned on. In Figures 17 and 18, the blue histogram counts the number of transactions that fall in each delay interval, and the red line graph calculates the proportion of the number of delays corresponding to the current interval in the total number of transactions, which we call it the cumulative proportion of delay in this interval is displayed on the ordinate on the right. From the two figures, we can see that after the sensor data encryption is turned on, it will affect the total number of transactions. As can be seen from Figure 18, the number of transaction delays in each interval has been reduced. However, after the sensor data encryption is turned on, the transaction delay time will not be affected. From the red line chart, we can see that all transaction delays are still below 5000 ms, and the cumulative proportion accounts for almost 100%.

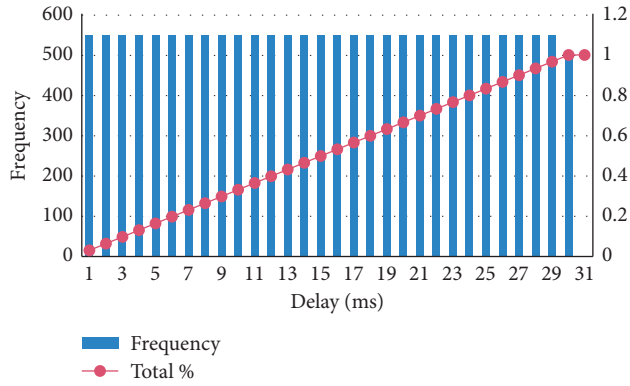


Figure 16: Transaction delay histogram (ring signature, $T = 30$ s).

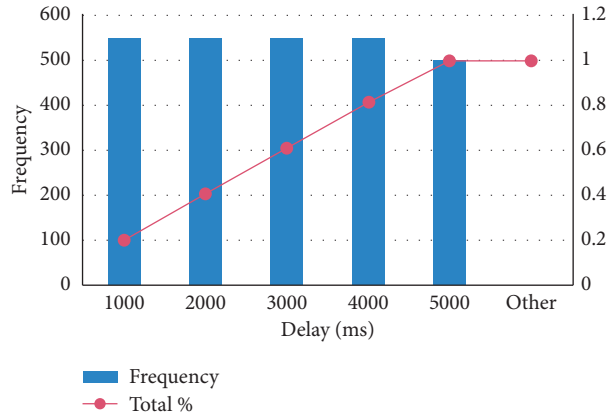


Figure 17: No sensor data encryption.

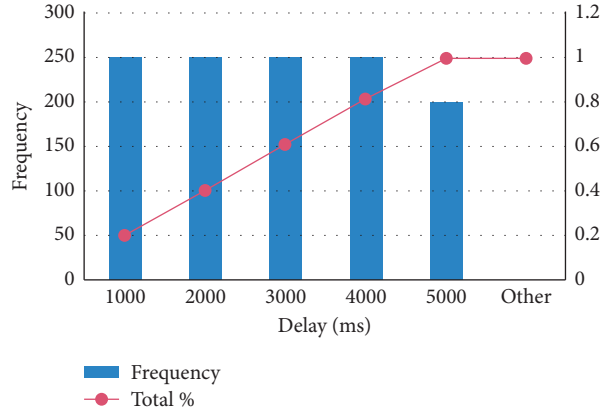


Figure 18: Enable sensor data encryption.

4. Conclusion

This article introduces sensor data privacy protection method for IoT based on blockchain technology. The ring signature realizes the anonymization of gateway transactions, prevents data sources from being tracked, and solves the anonymization problem of blockchain-based IoT users. Through asymmetric encryption, the sensor verifies the identity of the gateway and encrypts the sensor data. The gateway can create a block and submit a blockchain transaction after verifying the integrity and source of the data. Finally, combined with data access control and data encryption sharing, decentralized applications can finely control data access. Through experiments, the impact of transaction signatures on performance and the impact of sensor data encryption on performance are analyzed. The

results show that transaction delays are all controlled within a reasonable range. The system performance achieved by this method is also relatively stable.

Data Availability

The data that support the findings of the research are available from the corresponding author.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the Key Research and Development Plan of Anhui Province (201904a06020056). The authors are very grateful for the research foundation of the conference paper "IoT Data Privacy Protection Scheme Based on Blockchain" in IOP Science.

References

[1] M. A. Khan and K. Salah, "IoT security: review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[2] R. Mitchell and I.-R. Chen, "A survey of intrusion detection in wireless network applications," *Computer Communications*, vol. 42, pp. 1–23, 2014.

[3] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[4] A. A. Khan, M. H. Rehmani, and A. Rachedi, "Cognitive-radio-based internet of things: applications, architectures, spectrum related functionalities, and future research directions," *IEEE wireless communications*, vol. 24, no. 3, pp. 17–25, 2017.

[5] J. Granjal, E. Monteiro, and J. Sa Silva, "Security for the internet of things: a survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[6] S. M. H. Bamakan, N. Faregh, and A. ZareRavasan, "Di-ANFIS: an integrated blockchain-IoT-big data-enabled framework for evaluating service supply chain performance," *Journal of Computational Design and Engineering*, vol. 8, no. 2, pp. 676–690, 2021.

[7] S. Cirani, G. Ferrari, and L. Veltri, "Enforcing security mechanisms in the IP-based internet of things: an algorithmic overview," *Algorithms*, vol. 6, no. 2, pp. 197–226, 2013.

[8] Y. Liu, K. Wang, Y. Lin, and W. Xu, "Lightweight blockchain system for industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3571–3581, 2019.

[9] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with IoT. Challenges and opportunities," *Future Generation Computer Systems*, vol. 88, pp. 173–190, 2018.

[10] A. Dorri, S. S. Kanhere, R. Jurdak et al., "Blockchain for IoT security and privacy: the case study of a smart home," in *Proceedings of the IEEE Percom Workshop on Security Privacy and Trust in the Internet of Thing*, pp. 618–623, IEEE, Kona, HI, USA, March 2017.

[11] P. Patil, M. Sangeetha, and V. Bhaskar, "Blockchain for IoT access control, security and privacy: a review," *Wireless Personal Communications*, vol. 117, pp. 1–20, 2020.

[12] D. Minoli and B. Occhiogrosso, "Blockchain mechanisms for IoT security," *Internet of Things*, vol. 1-2, pp. 1–13, 2018.

[13] Y. Qian, Y. Jiang, J. Chen et al., "Towards decentralized IoT security enhancement: a blockchain approach," *Computers & Electrical Engineering*, vol. 72, pp. 266–273, 2018.

[14] R. Di Pietro, X. Salleras, M. Signorini et al., "A blockchain-based trust system for the internet of things," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, pp. 77–83, Indianapolis, IN, USA, June 2018.

[15] H. Si, C. Sun, Y. Li, H. Qiao, and L. Shi, "IoT information sharing security mechanism based on blockchain technology," *Future Generation Computer Systems*, vol. 101, pp. 1028–1040, 2019.

[16] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, 2001.

[17] N. P. Smart, "The exact security of ECIES in the generic group model," in *Proceedings of the IMA International Conference on Cryptography and Coding*, pp. 73–84, Springer, Cirencester, UK,, December 2001.

[18] L. Zhou, L. Wang, Y. Sun, and P. Lv, "BeeKeeper: a blockchain-based IoT system with secure storage and homomorphic computation," *IEEE Access*, vol. 6, pp. 43472–43488, 2018.