Research Article

Towards Time-Sensitive and Verifiable Data Aggregation for Mobile Crowdsensing

Tao Zhang ¹, Xiongfei Song ¹, Lele Zheng,¹ Yani Han,¹ Kai Zhang ² and Qi Li³

¹School of Computer Science and Technology, Xidian University, Xi'an 710071, China

²School of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710119, China

³School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China

Correspondence should be addressed to Xiongfei Song; xfsong@stu.xidian.edu.cn

Received 5 November 2020; Revised 10 December 2020; Accepted 7 January 2021; Published 22 January 2021

Academic Editor: Athanasios V. Vasilakos

Copyright © 2021 Tao Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Mobile crowdsensing systems use the extraction of valuable information from the data aggregation results of large-scale IoT devices to provide users with personalized services. Mobile crowdsensing combined with edge computing can improve service response speed, security, and reliability. However, previous research on data aggregation paid little attention to data verifiability and time sensitivity. In addition, existing edge-assisted data aggregation schemes do not support access control of large-scale devices. In this study, we propose a time-sensitive and verifiable data aggregation scheme (TSVA-CP-ABE) supporting access control for edge-assisted mobile crowdsensing. Specifically, in our scheme, we use attribute-based encryption for access control, where edge nodes can help IoT devices to calculate keys. Moreover, IoT devices can verify outsourced computing, and edge nodes can verify and filter aggregated data. Finally, the security of the proposed scheme is theoretically proved. The experimental results illustrate that our scheme outperforms traditional ones in both effectiveness and scalability under time-sensitive constraints.

1. Introduction

Recent years have witnessed the proliferation of smart devices in all areas of people's daily life. These devices are deployed in different Internet of Things (IoT) units, such as smartphones [1], smart cameras [2], wearable devices [3], and environmental sensors [4, 5]. The number of IoT devices is still growing rapidly in the near future. According to the prediction of the Global Association for Mobile Communications Systems [6], the number of IoT devices will reach 25.1 billion in 2025. The rapid growth of IoT devices has promoted the development of mobile crowdsensing. The massive IoT devices will together provide mobile crowdsensing systems with more real-time and high-precision data, from which we can extract valuable information for personalized service provision to the majority of end consumers [7]. However, many personalized services require low latency, which usually cannot be provided by the cloud because of the long distances. To address this challenge, edge computing was

proposed to achieve a cloud-side-end IoT architecture [8].

Edge computing can provide network, computing, and storage services between the cloud and terminal devices [9], which greatly reduce the response time [10]. Due to these advantages, edge computing combines technologies of IoT and cloud computing to provide more and better services [11]. For example, in the application of Internet of Vehicles [12, 13], the sharing of real-time data between a large number of vehicles and edge units improves the vehicle's environmental perception, decision-making, and execution capabilities [14]. However, IoT devices are vulnerable to be attacked. Some attacks can be resisted by traditional security mechanisms [15, 16]. But traditional security mechanisms are too expensive overhead to be widely used, especially for the resource-constrained IoT devices [17]. The rapid response capability of edge-assisted mobile crowdsensing can meet the requirements of low latency, but the employment of traditional security mechanisms will increase the communication delay of large-scale devices. In addition, traditional

data aggregation schemes cannot recover the original data. Failure to verify the data before aggregation leads to unreliable aggregation results. Consequently, some tough issues remain unsolved.

First of all, data aggregation requires a new mechanism to realize the key distribution of large-scale devices. Some existing aggregation schemes have strong security [18, 19], but they use traditional one-to-one encryption. These aggregation structures are complicated because the number of secret keys increases with the number of devices. These complex aggregation schemes are difficult to apply to IoT devices [20]. For example, service providers select candidate nodes to perform tasks, but the key management, encryption, and decryption of large-scale nodes will cause serious delays in data aggregation. Therefore, it is very important to use secure and efficient access control to realize the key distribution of large-scale devices to reduce latency.

Second, data verifiability has not been specifically considered in most previous data aggregation schemes. Usually, data are aggregated from IoT devices according to specific task requirements, such as sum aggregation or mean calculation. In the previous schemes [21, 22], the edge computing node was responsible for collecting and aggregating data, then sending the aggregation results to the cloud. The cloud can only obtain the aggregated result but cannot recover the original data. However, since the information provided by the device could be unreliable, the result of data aggregation may be inaccurate. If the original data cannot be recovered, the false data will be hidden in the aggregation results, leading to inaccurate services provided by the platform. Therefore, it is essential to recover and verify the original data from the aggregated results.

Finally, the time-sensitive data aggregation scheme can provide more accurate aggregation results. Personalized data aggregation tasks are usually time-sensitive. For example, we only need to count the data during the time period to monitor the traffic flow during the peak period; the weather forecast reports that the rainstorm level is also determined according to the rainfall in a certain period of time. Time sensitivity means that the task needs to be completed at a specific time. We set the age of the key to the time required by the task. However, existing data aggregation schemes [23, 24] without considering the time factor cannot meet the requirements of personalized services. Therefore, time parameters need to be taken into account for data aggregation.

Aiming at the above challenges, we propose a novel time-sensitive and verifiable data aggregation scheme for edge-assisted mobile crowdsensing. In our scheme, we use attribute-based encryption for access control, and only those authorized nodes can obtain the session key. The device uses the session key to encrypt data. When ciphertexts are collected by edge computing nodes, edge computing nodes preprocess and aggregate them, and then send the aggregation results to the platform. Finally, the platform recovers all the original data and digs out the required information from the legal data. Our contributions can be summarized as follows:

- (1) Efficient access control. The platform uses attribute-based encryption to achieve efficient access control. Because attribute encryption is a one-to-many encryption method, the platform can generate an access structure based on task attributes, and all legal users who meet this access structure can decrypt the session key.
- (2) Data verifiability. Edge computing nodes aggregate verified data, and the aggregated data can be restored in the cloud. Compared with the traditional non-verifiable solution, the data after verification can guarantee the reliability of data. The platform can provide more accurate services.
- (3) Time sensitivity. For personalized tasks with time requirements, edge computing nodes gather and upload encrypted data before the task time expires. After introducing time parameters, the accuracy of time-sensitive tasks can be improved.

1.1. Related Work. In the literature, many studies have been carried out on the security and efficiency of data aggregation. However, at present, most works do not take into consideration both the two aspects. Researchers use homomorphic encryption to ensure security, an encryption scheme that can operate in ciphertext, to achieve data aggregation. Wang et al. [25] used the Castagnos-Laguillaumie cryptosystem to realize data aggregation in fog computing. However, their work only supports summation and aggregation and cannot recover all the data. Lu et al. [26] proposed a data aggregation scheme to extend the aggregation function, in which the cloud server can calculate the mean and variance. However, the original data still cannot be recovered, and dynamic groups are not considered.

Another major problem with data aggregation is the existence of collusion attacks. To solve this problem, Shim and Park [27] proposed a homomorphic encryption-based data aggregation scheme based on heterogeneously encrypted WSN, which can resist certain attacks but cannot ensure data integrity. Shen et al. [28] adopted the identity-based cryptography. They proposed a secure WSN data aggregation method, which mainly solves collusion attacks, but its cost is very high.

To make data aggregation meet the requirements of time-sensitive tasks, some schemes introduce privacy protection mechanisms to achieve safe transmission. Since privacy protection guarantees the privacy of sensitive data during data aggregation and the cost is much lower than encryption, they have become a research hotspot. Zhang et al. [21] proposed a verifiable privacy protection aggregation scheme (PPAS) for urban sensor systems. However, it requires trusted hardware and other communications to protect data integrity. Later, Li et al. [22] proposed an effective mobile sensing PPAS, which adopted the idea of multisecret sharing. However, the existence of a trusted key trader and the adjustment of shares when users leave make it inefficient and inflexible.

To achieve verifiable data aggregation, Shen et al. [18] proposed a data aggregation scheme that supports fault tolerance and data integrity, but its inability to support outsourced calculations results in high node calculation overhead, and the scheme also requires a specific structure. To address this problem, Bao and Lu [29] proposed an aggregation scheme that supports dynamic groups, but this scheme needs to set a separate secret key for each device, which increases the cost of key management, encryption, and decryption.

In other aggregation schemes, such as the key management technology of [23], the sum of random numbers of all participants (including all users and control centers) is equal to 0 to reduce the authority of the control center. A major disadvantage of this mechanism is that they cannot tolerate any failures. Even if a single user cannot report data at a certain time, the control center will not obtain any information because the sum of the random numbers in the final encrypted aggregation is no longer zero. This may be a big problem, and there is no guarantee that all devices will not fail during data aggregation.

1.2. Organization. The remainder of this study is organized as follows. Preliminaries are described in Section 2. Next, the system model and security definitions are presented in Section 3. In Section 4, we construct the TSVA-CP-ABE scheme. Security analysis and performance evaluation of the TSVA-CP-ABE are presented in Sections 5 and 6, respectively. Finally, we conclude our work in Section 7.

2. Background

In this section, we give the background information of bilinear maps, access structure, and Lagrange coefficient which will be used in our proposed scheme.

2.1. Bilinear Maps

Definition 1 (Bilinear maps). Let G and G_T be two multiplicative cyclic groups with big prime order p . Let g be a generator of G . Let e be a bilinear map $e: G \times G \rightarrow G_T$ with the following properties:

Bilinearity: for $\forall u, v \in G$ and $a, b \in \mathbb{Z}_p$, the equation $e(u^a, v^b) = e(u, v)^{ab}$ holds

Nondegeneracy: $e(g, g) \neq 1$

Computability: there exists an efficient algorithm to compute the bilinear map $e: G \times G \rightarrow G_T$

2.2. Access Structure. In our construction, we represent the access structure through an access tree Γ with root r . A leaf node of the tree represents an attribute, and a nonleaf node represents a threshold gate. If the node x is a leaf node, set $\Gamma_x(S) = 1$ if and only if the attribute of the node x match the attribute set S . If x is a nonleaf, $\Gamma_x(S) = 1$, if and only if the number of children nodes that meet the requirements is more than the threshold.

Definition 2 (Access structure [30]). Let $\{P_1, \dots, P_n\}$ be a set of parties. A collection $\mathbb{A} \in 2^{P_1, \dots, P_n}$ is monotone if $\forall B, C$ if $B \in \mathbb{A}$, and $B \subseteq C$, then $C \in \mathbb{A}$. An (monotone) access structure is a (monotone) collection $\mathbb{A}$ of nonempty subsets $\{P_1, \dots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{P_1, \dots, P_n} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets; otherwise, they are called unauthorized sets.

2.3. Lagrange Coefficient. In our scheme, we use the Lagrange coefficient to achieve a collision-resistant secret sharing.

Definition 3 (Lagrange coefficient). The Lagrange coefficient $\Delta_{i,S}$ for $i \in \mathbb{Z}_p$ and a set, S , of elements in $\mathbb{Z}_p$: $\Delta_{i,S}(x) = \prod_{j \in S, j \neq i} ((x - i)/(i - j))$. We will associate each attribute with a unique element in $\mathbb{Z}_p^*$.

3. System Model and Definitions

In this section, we first describe the system model. Next, we present replayable chosen-ciphertext attack security of the TSVA-CP-ABE scheme. Finally, we formulate the efficient data aggregation problem.

3.1. System Model. In this study, we consider the typical IoT data aggregation system, as shown in Figure 1. The system consists of four kinds of entities, namely, IoT devices, edge computing nodes, a mobile crowdsensing platform, and a trusted authority [31]. More specifically, sensory data are first uploaded from the IoT devices to the edge computing node, processed, and relayed to the mobile crowdsensing platform for further aggregation to obtain more valuable results. The data sensed by devices can be of any format, such as big string, video, or pictures. We assume that there are two-way communication channels between different entities in our system, where each entity can upload and download necessary data. More specifically, the role of each entity is described as follows:

- (1) Trusted authority (TA). TA initializes the system by publishing the public parameter (MPK, MSK) and provides the attribute-related private keys (DK, TK_{T_i}) for ID. TA accepts the registration of all IDs in the system, and it is responsible for issuing a pseudonym ID pid for each ID. TA stops updating attacked IDs' private keys by pid.
- (2) Mobile crowdsensing platform (MCP). When encrypting, MCP encrypts the time-sensitive task as published ciphertext CT_{T_i} to the corresponding ID and ECN. When decrypting, MCP uses the secret key corresponding to the task to decrypt the aggregated data c' . And, MCP judges the legality of data and stores the legal data.
- (3) Edge computing node (ECN). ECN responds to outsource decryption to generate TCT_{T_i} from legality nodes and filter out expired ciphertexts. After that, ECN verifies the legal of encrypted data and aggregates the legal data and then forwards c' to

FIGURE 1: System model of the TSVA-CP-ABE.

MCP. In the decryption and data aggregation, ECN cannot decrypt the ciphertext.

- (4) IoT device (ID). ID uses their own private key associated with its attributes to decrypt the message to obtain the symmetric key and then uses the symmetric key to encrypt the uploaded data c .

In our scheme, TA and MCP are trustworthy. ECN is assumed to be honest-but-curious [32]. In the decryption, ECN only transforms ciphertext. In data aggregation, ECN can only verify the reliability of data. It honestly executes the specified procedures but tries to gain secret information from the encrypted data. The adversary could be a malicious ID or a group of multiple IDs, which may post false information and try to control other normal IDs. Moreover, the adversary also aims to obtain the symmetric key from attribute-based encryption. Therefore, TA should be able to revoke the keys from malicious IDs. Actually, the assumption of this model is widely used in previous work [18, 19, 21, 22].

Definition 4 (Syntax of the TSVA-CP-ABE). A CP-ABE system with verifiable outsourced decryption for time-sensitive tasks consists of the following probabilistic polynomial time (PPT) algorithms:

Setup $() \rightarrow (MPK, MSK)$: the setup algorithm takes no input other than the implicit security parameter. It outputs a master public key MPK and a master secret key MSK . Trusted authority publishes the parameters and keeps MSK secret to itself.

Encrypt $(MPK, M, \gamma, T_s) \rightarrow (CT_{T_s}, VT2)$: the encrypt algorithm takes as input a public key MPK , a shared message M which the owner wants to encrypt, an access

tree γ , and a task of time T_s . It outputs a ciphertext CT_{T_s} which is associated with a task of time T_s and a verification tag $VT2$.

KeyGen $(MSK, S, T_c) \rightarrow (TK_{T_c}, DK)$: the KeyGen algorithm takes as input a master secret key MSK , an attribute set S , and the current time T_c . It outputs a decryption key DK and a transformation key TK_{T_c} which is associated with the current time T_c .

Transform $(TK_{T_c}, CT_{T_s}, VT2) \rightarrow (TCT_{T_s})$: the transform algorithm takes as input a transformation key TK_{T_c} , a ciphertext CT_{T_s} , and a verification tag $VT2$. It outputs a transformation decrypted ciphertext TCT_{T_s} which is associated with a task of time T_s .

Decrypt $(DK, VT1, TCT_{T_s}) \rightarrow (M/\perp)$: the decrypt algorithm takes as input a decryption key DK_{T_c} , a verification tag $VT1$, and a transformation decrypted ciphertext TCT_{T_s} . It outputs a message $M/\perp$. Here, the special symbol $\perp$ indicates that the transformation decrypted ciphertext is invalid.

3.2. RCCA Security. The security of our scheme is based on the replayable chosen-ciphertext attack security (RCCA) [33] for ABE with verifiable outsourced decryption and verification keys. Different from their schemes, we add the time parameter, which also increases the additional security risk. The security of the time parameter will be proved later.

(i) **Setup.** The challenger runs the $Setup(k, U)$ to generate the master key pair (MPK, MSK) . It gives MPK to the adversary.

(ii) **Phase 1.** The challenger initializes an empty table E , an empty set D , and a counter $j = 0$. Proceeding

adaptively, the adversary can repeatedly make any of the following queries:

Create (S, T_c) : the challenger sets $j = j + 1$. It first runs $\text{KeyGen}(\text{MSK}, S, T_c)$ to obtain a key pair $(\text{TK}_{T_c}, \text{DK})$. Then, it returns TK_{T_c} to the adversary and stores in table E the entry $(j, S, T_c, \text{TK}_{T_c}, \text{DK})$.

Corrupt (i) : if there exists an i entry in table E , the challenger obtains the entry $(i, S, T_c, \text{TK}_{T_c}, \text{DK})$. If $f(\gamma, S) = 1$, it returns $\perp$; otherwise, it returns the decryption key DK to the adversary and sets $D = D \cup \{S\}$.

Transformation $(i, \text{CT}, \text{VT2}, t)$: if there exists an i entry in table E . Meanwhile, both VT2 and t are valid; then, the challenger obtains the entry $(i, S, T_c, \text{TK}_{T_c}, \text{DK})$ and returns the output of (TCT, DK) to the adversary.

(iii) Challenge. The adversary submits two equal-length messages M_0 and M_1 as well as a challenge access tree γ^* such that for all $S \in D$, $f(S, \gamma^*) \neq 1$. The challenger flips a random coin b and encrypts M_b under γ . The ciphertext CT^* and verification key VT2^* are given to the adversary.

(iv) Phase 2. The same as phase 1 except that

All the ciphertexts must be decrypted by outsourced; otherwise, the correct plaintexts cannot be decrypted. If a decryption response would be either M_0 or M_1 , then the challenger responds with the special message $\perp$.

(v) Guess. The adversary outputs a guess b' of b .

The advantage of $\mathcal{A}$ in this game is defined as $\text{Adv} = |\Pr[b = b'] - 1/2|$.

Definition 5 (RCCA security). The TSVA-CP-ABE is RCCA-secure, if for any PPT adversary $\mathcal{A}$ has at most a negligible advantage in the above game.

3.3. Verifiable Outsourced

Setup. The challenger runs the $\text{Setup}(k, U)$ to generate the master key pair (MPK, MSK) . It gives MPK to the adversary.

Phase 1. The adversary can adaptively query the create, corrupt, and transformation oracles as in phase 1 in Section 3.2.

Challenge. The adversary submits a message M^* and an access tree γ^* . The challenger computes a challenge ciphertext CT^* and sends it to $\mathcal{A}$.

Phase 2. The same as phase 1.

Output. Finally, the adversary outputs a set S^* such that $f(\gamma^*, S^*) = 1$ and a transformation ciphertext TCT .

Above presents a formal definition of the verifiability for an ABE system with outsourced decryption, through a game played between an adversary $\mathcal{A}$ and a challenger.

Suppose that the entry $(j, S^*, T_c^*, \text{TK}_{T_c}^*, \text{DK}^*)$ has already existed in table E . If not, the challenger can generate it using

MSK . We say that $\mathcal{A}$ succeeds in the above defined game, if $\text{Decrypt}(\text{DK}^*, \text{VT1}^*, \text{TCT}^*) \notin \{M^*, \perp\}$. $\mathcal{A}$'s advantage is defined as $\text{Adv} = \Pr(\mathcal{A} \text{ succeeds in the above game})$.

Definition 6 (Verifiability). An ABE system with outsourced decryption is verifiable, if $\forall$ PPT adversary $\mathcal{A}$ has at most a negligible advantage in the above game [31].

3.4. Efficient Access Control. In our scheme, we assume all parties hold the same secret key for each task. We use attribute-based encryption for access control, and authorized nodes can obtain session keys.

Phase 1. MCP uses a symmetric secret key to encrypt the data to be sent. MCP selects the corresponding attribute encryption symmetric key according to the task requirements.

Phase 2. ECN is responsible for outsourcing and decrypting attribute-based encryption to reduce the computing burden of IDs.

Phase 3. ID decrypts the symmetric key according to its own attributes and then uses the symmetric key to communicate with MCP.

4. Construction of the TSVA-CP-ABE Scheme

In this section, we will introduce our scheme in two levels: system level and algorithm level. The system level describes the implementations of the upper operations, while the algorithm level focuses on the specific details of the underlying algorithms called by system level operations [34]. We depict the framework of our scheme in Figure 2. The details of these two levels are described as follows.

4.1. System Descriptions

4.1.1. System Setup. In the system initialization phase, TA invokes the algorithm $\text{Setup}()$ belonging to the algorithm level to generate master public key MPK and master secret key MSK . MCP obtains MSK and shares a random number and a key with ECN in a secure way (such as public key encryption) at the same time. When IDs register with TA, TA uses a one-way function to generate a key-value pair based on the device ID did , where the key is did , and the value is pid . ID uses a pseudonym ID when submitting data to MCP, thereby protecting the privacy of ID. If MCP wants to revoke a device, it provides the pseudonym ID to TA. Then, TA will revoke the real id.

4.1.2. Time-Sensitive Encryption. The tasks issued by MCP are time-sensitive. MCP chooses an access tree γ for the message M and defines a time interval set T_s for ciphertext. Then, MCP invokes the algorithm $\text{Encrypt}(\text{MPK}, M, \gamma, T_s)$ to generate ciphertext CT_{T_s} , which is associated with γ and T_s . Finally, CT_{T_s} is sent to IDs. ID will be able to decrypt a ciphertext if and only if ID's attributes satisfy the access tree and ID's key is valid for a limited period of time.

FIGURE 2: The framework of generic construction of the TSVA-CP-ABE.

4.1.3. Access Control. Attribute-based encryption is a one-to-many encryption scheme. When MCP issues a task, it constructs an access structure based on attributes of the task performer. Any ID that satisfies the access structure can decrypt ciphertext according to its own private key and obtain the session key of the corresponding task. Thus, MCP achieves key distribution in this way.

4.1.4. Selective Outsourced. ID applies for a decryption key from TA. TA invokes the algorithm KeyGen (MSK, S, T_c) to generate a decryption key DK and a transformation key TK_{T_c} which is associated with the current time interval T_c . Then, ID sends the ciphertext CT_{T_s} and the transformation key TK_{T_c} to ECN to apply for outsourced decryption. After receiving the outsourced decryption application, ECN does not directly decrypt it. It first checks whether the ID is in a white list, which stores IDs that have not been revoked. If ID exists in a white list, then it verifies the validity of the ciphertext by the random s_2 previously shared with the platform ($H(s_2, CT_{T_s}) = VT2$) and verifies the validity for a limited period time of the secret key ($T_c + T_{all} < T_s$). Then, ECN conducts the outsourced decryption after both verifications pass.

4.2. Algorithm Constructions. Here, we will give the concrete constructions of the TSVA-CP-ABE scheme. The notations used in the TSVA-CP-ABE scheme are given in Table 1.

(i) Setup. The setup algorithm will choose a bilinear group $\mathbb{G}$ of prime order p with generator g . Next, it will choose two random exponents $\alpha, \beta \in \mathbb{Z}_p$. In addition,

let T be the maximum time in the system. T is provided by TA which satisfies $|T| = n$. Then, we choose $u_1, \dots, u_n \in \mathbb{Z}_p$ randomly. The master public key is published as

$$MPK = \mathbb{G}, g, h = g^\beta, e(g, g)^\alpha, \quad u_1, \dots, u_n \in \mathbb{Z}_p, \quad (1)$$

and the master secret key MSK is (β, g^α) .

(ii) Encrypt (MPK, M, γ, T_s). We will additionally employ two hash functions $H: \{0, 1\}^* \rightarrow \mathbb{G}, H_0: M \rightarrow \{0, 1\}^{l_{H_0}}$. The encryption algorithm encrypts a message M ($M = k_1 \| k_2$) under the tree access structure γ . The algorithm first chooses a polynomial q_x for each node x (including the leaves) in the tree γ . These polynomials are chosen in the following way in a top-down manner, starting from the root node R . For each node x in the tree, we set the degree d_x of the polynomial q_x to be one less than the threshold value k_x of that node, that is, $d_x = k_x - 1$.

Starting with the root node R , the algorithm chooses a random $s_1 \in \mathbb{Z}_p$ and sets $q_R(0) = s_1$. Then, it chooses d_R other points of the polynomial q_R randomly to define it completely. For any other node x , it sets $q_x(0) = q_{\text{parent}(x)}(\text{index}(x))$ and chooses d_x other points randomly to completely define q_x .

MCP generates a random s_2 and sends it safely to ECN. MCP also generates verification tags VT1 and VT2. Let Y be the set of leaf nodes in γ . The ciphertext constrained by a time interval T_s is then constructed by giving the tree access structure γ and computing

$$\begin{aligned}
CT_{T_s} &= (\tilde{C} = \text{Me}(g, g)^{\alpha s_1 s_2 L_{T_s}}, \\
C &= h^{s_1 L_{T_s}}, \\
VT1 &= H_0(M), \gamma, T_s, \forall y \in Y: C_y = g^{q_y(0)}, \\
C'_y &= H(\text{att}(y))^{q_y(0)}, \\
VT2 &= H_0(s_2, CT_{T_s}), \\
L_{T_s} &= \sum_{k=1}^{T_s} u_k^{t_k}.
\end{aligned} \tag{2}$$

(iii) KeyGen (MSK, S, T_c). The key generation algorithm will take as input a set of attributes S , MSK, and the current time interval T_c . It will output a decryption key DK and a transformation key TK_{T_c} which is associated with the current time interval T_c . The algorithm first chooses a random $r \in \mathbb{Z}_p$ and then random $r_j \in \mathbb{Z}_p$ for each attribute $j \in S$. Then, it computes the key as

$$\begin{aligned}
\text{TK}_{T_c} &= \left(D = g^{(\alpha/(\alpha'+r))/\beta}, \left\{ \forall k = T_c + 1: n, g^{ru_k^{t_k}} \right\} \forall j \in S: D_j = g^{rL_{T_c}} \cdot H(j)^{r_j}, D'_j = g^{r_j} \right), \\
\text{DK} &= \alpha'.
\end{aligned} \tag{3}$$

(iv) Transform ($\text{TK}_{T_c}, \text{CT}_{T_s}, \text{VT2}$). When ECN receives the request of outsourcing decryption, ECN first checks whether the ID is in a white list. If ID exists in a white list, then ECN verifies the legality of the message with the random s_2 agreed with the platform and makes a judgment on the time parameters to decide whether to perform outsourced. If $\text{VT2} = \text{VT2}'$ ($\text{VT2}' = H(s_2, \text{CT}_{T_s})$) and $T_c + T_{\text{all}} < T_s$, it starts to perform outsourced calculation.

We first define a recursive algorithm TransformDecrypt ($\text{TK}_{T_c}, \text{CT}_{T_s}, x$) that takes as input a ciphertext CT_{T_s} , a node x belongs to γ , and a transformation key TK_{T_c} . TK_{T_c} is associated with a set S of attributes and the current time interval T_c . Before calculating, we define L as the sum of time parameters from T_c to T_s .

If the node x is a leaf node, then we let $i = \text{att}(x)$ and define as follows: if $i \in S$, then

$$\begin{aligned}
L &= \prod_{k=T_c+1}^{T_s} g^{ru_k^{t_k}}, \\
\text{TransformDecrypt} &= \frac{e(D_i L, C_x)}{e(D'_i, C'_x)} = \frac{e(g^{rT_s} \cdot H(i)^{r_i}, g^{q_x(0)})}{e(g^{r_i}, H(i)^{q_x(0)})} = e(g, t g)^{rL_{T_s} q_x(0)}.
\end{aligned} \tag{4}$$

If $i \notin S$, then we define $\text{TransformDecrypt}(\text{TK}_{T_c}, \text{CT}_{T_s}, x) = \perp$.

We consider the recursive case when x is a nonleaf node. The algorithm TransformDecrypt ($\text{TK}_{T_c}, \text{CT}_{T_s}, x$) then proceeds as follows: for all nodes z that are children of x , it invokes TransformDecrypt

($\text{TK}_{T_c}, \text{CT}_{T_s}, z$) and stores the output as F_z . Let S_x be an arbitrary k_x -sized set of child nodes z such that $F_z \neq \perp$. If no such set exists, then the node was not satisfied and the function outputs $\perp$.

Otherwise, we compute

$$\begin{aligned}
F_x &= \prod_{z \in S_x} F_z^{\Delta_{iS'_x}(0)}, \quad \text{where } i = \text{index}(z), S'_x = \{\text{index}(z): z \in S_x\} = \prod_{z \in S_x} (e(g, g)^{rL_{T_s} q_z(0)})^{\Delta_{iS'_x}(0)} \\
&= \prod_{z \in S_x} (e(g, g)^{rL_{T_s} q_{\text{parent}(z)}(\text{index}(z))})^{\Delta_{iS'_x}(0)} = \prod_{z \in S_x} e(g, g)^{rL_{T_s} q_x(i) \Delta_{iS'_x}(0)} = e(g, g)^{rL_{T_s} q_x(0)}.
\end{aligned} \tag{5}$$

TABLE 1: Notations.

Symbol	Description
k_1	The secret key of the session between MCP and the ID
k_2	The authentication key
s_1	The random number added in ciphertext
s_2	The shared random number between MCP and ECN
T_c	The key generation time
T_s	The ciphertext generation time
T	The system period
t_i	The time of the i^{th} interval
H, H_0	Two collision-resistant hash functions
VT1, VT2	The verification tag
T_1, T_2	The time of data aggregation
T_{all}	The total time of data transfer and computation
MPK, MSK	The master public key and the master secret key
M	The shared message

The algorithm begins by simply calling the function on the root node R of the tree γ . If the tree is satisfied by S , we set $A = e(g, g)^{rL_{T_s}s_1}$. The algorithm now outputs transformed ciphertext by computing

$$\frac{e(C, D)}{A} = \frac{e(h^{s_1 L_{T_s}}, g^{(\alpha/\alpha' + r)/\beta})}{e(g, g)^{rL_{T_s}s_1}} = e(g, tg)^{L_{T_s}s_1\alpha/\alpha'},$$

$$\left(\frac{e(C, D)}{A}\right)^{s_2} = e(g, g)^{L_{T_s}s_1s_2\alpha/\alpha'}.$$
(6)

(v) Decrypt (DK, VT1, TCT $_{T_s}$). The decryption algorithm will take as input TCT $_{T_s}$, a decryption key DK, and a verification tag VT1. It will recover the message M^* . If VT1 = VT1' (VT1' = $H_0(M^*)$), the decryption is successful. Otherwise, the decryption fails. The result can be verified as follows:

$$\frac{\tilde{C}}{e(g, g)^{(L_{T_s}s_1s_2\alpha/\alpha')\alpha'}} = M. \quad (7)$$

(vi) Data aggregation (c, ct). When ID completes the final decryption, it obtains the session key, including k_1 (encryption key) and k_2 (authentication key). First, ID uses k_1 to complete the data encryption $c = e(\text{data}, k_1)$ (e is symmetric encryption) and then generates $ct = c \| H_0(k_2, T_1, c) \| T_1$, where T_1 is the current time interval when ID submitted data. Second, ECN receives the data and verifies the time and hash value. Once the verification is passed, the redundant data will be removed. Only the ciphertext is aggregated

to generate c' , and then, c' is used to generate $ct' = c' \| H_0(k_2, T_2, c') \| T_2$, where T_2 is the current time interval when ECN submitted data. Third, MCP receives the data and uses the same method to verify ciphertext and removes redundant data after verification. All ciphertexts are aggregated to generate c'' . Finally, MCP uses the encryption key k_1 to decrypt all data for the task.

5. Security of the Scheme

In this section, we detail the security proof of our scheme.

5.1. Security of Time Parameters. The master public key is published as $\text{MPK} = \mathbb{G}, g, h = g^\beta, e(g, g)^\alpha, u_1, \dots, u_n \in \mathbb{Z}_p$, which makes the parameters of the time interval in the group $u_1, \dots, u_n$ unsafe. The adversary can counterfeit or copy the parameters of the time interval because it obtains additional information from MPK. First, in order to ensure the unforgeability of u_n , we add a random number s_1 whenever we encrypt. In ciphertexts, the time parameter we add is $s_1 L_{T_s} = s_1 \sum_{k=1}^{T_s} u_k^{t_k}$. Then, in order to ensure the freshness of the secret key, we add the current time parameter t_i to u_i . For the parameters in key generation, we add $rL_{T_c} = r \sum_{k=1}^{T_c} u_k^{t_k}$ to the secret key and send $\{ \forall k = T_c + 1: n, g^{ru_k^{t_k}} \}$ to the decryptor. In combination with the difficult problem (discrete logarithm problem [35]: in group $\mathbb{G}$, given generators g and $a \in \mathbb{Z}_1$, it is easy to find g^a . When given g^a and generators g , it is difficult to calculate a), if c is smaller than or equal to s , the ciphertext can be decrypted. Otherwise, the ciphertext cannot be decrypted.

$$L_{T_c} = \sum_{k=1}^{T_c} u_k^{t_k}, \quad \left\{ \forall k = T_c + 1: n, g^{ru_k^{t_k}} \right\} \left\{ \begin{array}{ll} L_{T_s} = L_{T_c} \prod_{k=T_c+1}^{T_s} g^{ru_k^{t_k}}, & c \leq s, \\ L_{T_s} = \perp, & c > s. \end{array} \right. \quad (8)$$

5.2. Security of the TSVA-CP-ABE

Theorem 1. *Our ABE system with verifiable outsourced decryption is (selectively) CPA-secure if and only if the underlying outsourced ABE system is (selectively) CPA-secure. H_0 is a collision-resistance hash function, and SE is a semantically secure one-time symmetric encryption scheme.*

Proof. The proof applies the hybrid argument of games. We define two games: Game_0 and Game_1 . Game_0 is the original (selective) CPA-security game as defined in Section 3.2 for an outsourced ABE system. We intend to prove that any two subsequent games only have a negligible difference from the adversary's perspective. Let S_i denote $\mathcal{A}$'s success probability in Game_i . Above, we have proven the security of time parameters. When the time parameter is not satisfied, it cannot be decrypted. Therefore, we assume that the time parameters are valid in the following proof process. On this basis, we prove the security of the proposed scheme.

Game_0 : this is the original (selective) CPA-security game. Let $(\text{CT}_T^*, \text{VT}2^*)$ denote the challenge ciphertext and verification tag for a challenge access tree γ selected by the adversary. Denote by $M = k1\|k2$, the key encrypted in ciphertext, and by $k1$, the symmetric key used in ciphertext in the uploading data.

Game_1 : this game is the same as Game_0 , except that we compute $\text{VK}1^*$ and $\text{VK}2^*$ using other random keys $k3$ and $k4$ and number s_2^* $\square$

Claim 1. Suppose that the outsourced ABE system is (selectively) CPA-secure, then the adversary's views in Game_0 and Game_1 are computationally indistinguishable.

Proof of Claim 1. We define a PPT algorithm $\mathcal{B}$ which aims to break the (selective) CPA-security of the underlying ABE system under the help of the adversary $\mathcal{A}$. $\mathcal{B}$ simulates $\mathcal{A}$'s views in Game_0 or in Game_1 depending on its challenge ciphertext. Denote by Chall , the challenger of the underlying ABE system.

Setup. $\mathcal{B}$ first runs Chall to obtain a challenge public parameter MPK^* . Then, it chooses by itself a collision-resistant hash function H_0^* and a semantically secure one-time encryption scheme SE^* . Finally, it sends $(\text{MPK}^*, H_0^*, \text{SE}^*)$ to $\mathcal{A}$ as a challenge master public key.

Phase 1. It is straightforward to answer $\mathcal{A}$'s queries, including create (S, T_c) (for any attributes S) and corrupt (i). This is because $\mathcal{B}$ can obtain the answers of these queries via running Chall with the same queries.

Challenge. Once $\mathcal{A}$ submits two equal-length messages M_0 and M_1 as well as an access tree γ^* , the simulator $\mathcal{B}$ first chooses four independent random keys $k1\|k2, k3\|k4$. It then queries Chall with $((k1\|k2, k3\|k4), \gamma^*)$. Chall will return a challenge ciphertext CT_{T_s} to $\mathcal{B}$. Next, $\mathcal{B}$ sets $\text{VK}1 = H_0^*(k1\|k2)$. It also computes $C_{\text{SE}}^* = \text{SE}^*(k1, M_b)$ for a random

$b \in \{0, 1\}$ and sets $\text{VK}2 = H_0^*(s_2, \text{CT}_{T_s})$. Finally, it sends CT_{T_s} and $\text{VK}2$ to the adversary. Clearly, if the "message" encrypted in CT_{T_s} is $k1\|k2$, then CT_{T_s} is a challenge ciphertext as in Game_0 . Otherwise, it is a challenge ciphertext as in Game_1 .

Phase 2. The same as phase 1 except that $\mathcal{A}$ cannot query corrupt (i), in which the attribute S satisfies $f(\gamma^*, S) = 1$.

Finally, $\mathcal{B}$ outputs what $\mathcal{A}$ outputs. From the above analysis, $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s views in Game_0 or Game_1 . So, we have the following result:

$$|\Pr[S_0] - \Pr[S_1]| \leq \text{Adv}_{\text{ABE}}. \quad (9) \quad \square$$

Claim 2. Suppose that the symmetric encryption scheme SE is semantically secured, then the adversary in Game_1 has a negligible advantage.

Proof of Claim 2. The security of Claim 2 depends on the symmetric cipher scheme we choose. Here, we choose the AES (advanced encryption standard) scheme that has proven to be secure. Therefore, the security of Claim 2 is guaranteed.

Taking all the claims together, the (selective) CPA-security of our scheme is given as follows.

In the above security proof, we only consider the (selective) CPA-security of our scheme. Similarly, we can prove its RCCA-security if the underlying outsourced ABE scheme is RCCA-secure and the symmetric encryption scheme is also RCCA-secure. $\square$

Claim 3. Suppose that H_0 is a collision-resistant hash function. Then, the ABE scheme we proposed with outsourced decryption is verifiable.

Proof. Given an adversary $\mathcal{A}$ against the verifiability, we construct an efficient algorithm $\mathcal{B}$ to break the collision-resistance of the underlying hash function H_0 . Given a challenge hash function H_0^* , $\mathcal{B}$ simulates the experiment described in Section 3.3 as follows.

$\mathcal{B}$ generates the public parameter MPK and master secret key MSK as setup, except for a hash function H_0^* . Note that, $\mathcal{B}$ knows the master secret key MSK . Hence, it can simulate $\mathcal{A}$'s queries in phase 1 and phase 2. For a challenge message M^* and an access tree γ^* submitted by $\mathcal{A}$, the simulator first invokes $\text{encrypt}(\text{MPK}, M^*, \gamma^*, T_s)$ to obtain a ciphertext $\text{CT}_{T_s}^*$. It then sets $\text{VT}1^* = H_0(M^*)$ and $\text{VT}2^* = H_0(s_2, \text{CT}_{T_s}^*)$. After that, it sends $\text{CT}_{T_s}^*$ and $\text{VT}2^*$ to the adversary. Finally, the adversary outputs attributes S^* (such that $f(\gamma^*, S^*) = 1$) and a transformation ciphertext TCT_{T_s} . If $\mathcal{A}$ breaks the verifiability, $\mathcal{B}$ will recover a message $M \notin \{M^*, \perp\}$ via decrypt. Here, we discuss $\mathcal{A}$'s success probability. Observe that the decryption algorithm outputs $\perp$ if $H_0^*(M) \neq \text{VT}1^*$. Therefore, we only need to consider the following the case:

Case: $(s_2, CT_{T_s}) = (s_2, CT_{T_s}^*)$, but $M \neq M^*$. Observe that $H_0^*(M) = H_0^*(M^*)$. Therefore, it breaks the collision-resistance of H_0^* . $\square$

5.3. Resist DDOS. The attacker can forge a large amount of false data and request ECN to perform outsourced decryption. When ECN processes false data, it cannot provide services to normal IDs. In our scheme, ECN will verify the data before calculation, and only the data that passes the verification will be calculated. The key and random number shared between ECN and MCP are included in the verification. The attacker cannot obtain these two parameters and therefore cannot forge data that can be verified by the edge computing node. Therefore, our scheme can resist distributed denial of service attacks.

5.4. Resist Replay Attacks. The attacker can capture a large amount of normal data and send a large amount of expired normal data at a later time period, thereby consuming the computing resources of MCP and ECN. In our proposed scheme, all data will be added with the timestamp, and the timestamp is protected by a hash function with a secret key. If the timestamp is modified, the hash value will return a different result. Thus, our scheme can protect the freshness of the data.

5.5. Revocation Security. Assuming that a certain ID has revoked its decryption capability, it will not continue decrypting new ciphertexts. In our scheme, ECN maintains a white list and only performs outsourced decryption for IDs in the white list. ECN and MCP share the same random number. When MCP encrypts tasks, random numbers are embedded in the ciphertext. ID cannot obtain the plaintext without outsourced decryption. ID that is not on the white list cannot be outsourced, which is equivalent to ID being revoked. This scheme makes the revoked ID unable to decrypt without affecting other legal IDs.

6. Analysis of the TSVA-CP-ABE Scheme

In this section, we first describe the theoretical analysis and the comprehensive comparison. Then, we present the experimental analysis of the TSVA-CP-ABE scheme.

6.1. Theoretical Analysis. We first compare several related works theoretically. Table 2 gives a comparison of the results of our work and several related works in terms of features. The compared schemes all support data integrity. However, Fan et al.'s scheme [23] is a summation aggregation, which cannot detect the legality of single data. Therefore, it does not support data fault tolerance. Moreover, schemes in [36] and in [23] must determine the number of IDs, and the aggregation can be successful only after all IDs have uploaded data. Therefore, they cannot support the dynamic joining and exiting of some devices. Our proposed scheme adds verification to ECN, and we can thus choose

outsourced verification, which is not supported by other schemes [18, 23, 36].

In this analysis, we focus on the most time-consuming operations, paring, and exponentiation conducted in groups G . Let ψ , ω , and ν , respectively, denote computation times of the most time-consuming operations, paring, and exponentiation. Let $O(\text{tree})$ be the computation complexity of the decryption tree. In the encryption stage, when the ciphertext is generated, a pair is made, and then, two ciphertext parameters, time parameters, and access structure parameters are exponentially calculated. The total time cost is $\psi + \omega + (|T| + 2 + |S|)\nu$. In the KeyGen stage, both the time parameter and the secret key parameter are exponential operations, and the properties in the secret key are paired operations. The total time cost is $\psi + |Y|\omega + (n + 1)\nu$. In the transform stage, when ECN decrypts, it pairs the secret key and the access structure and then performs an exponential operation on the structure with its own parameters. The total time cost is $O(\text{tree}) + |Y|\omega + \nu$. In the decryption stage, ID needs to do exponential operation on the time parameter. Finally, it can do an exponential operation on the ciphertext again. The total time cost is $\psi + n\nu$.

Different from the above works, we propose a time-sensitive and verifiable data aggregation scheme supporting access control for edge-assisted mobile crowdsensing. Our scheme is compared with the schemes in [18, 23] in Table 3. In our scheme, though pair operations, exponentiations, and multiplications are explored, these operations are only used to generate or verify the signatures. Moreover, any secure aggregate signature scheme can be utilized in our scheme. In [18], the time parameter is introduced in the scheme to make the secret key have a deadline. But it does not support outsourcing, which leads to high computational cost. In [23], only the sum of the sensing data can be recovered by the cloud. Comparatively, in our scheme, MCP can collect all the raw data and compute any statistical function on them. Besides, our scheme can resist collision attacks and support dynamic groups.

6.2. Experimental Analysis. We simulate our scheme on a laptop with an Intel Core i5-3210M CPU at 2.50 GHz and 8 GB RAM running on Eclipse 4.10 and Windows 7. Charm-crypto framework integrated with the OpenSSL and JPBC library is applied to implement the cryptographic operations. Besides, the group operations are based on the elliptic curve of SS512, and the number of policy attributes associated with ciphertext is from 10 to 50. The simulation results are averaged on 10 independent runs. Since we use a computer to simulate the whole communication process, we give the test values of bandwidth and network delay in the communication process. MCP, TA, and ECN are connected via wire networks. ECN, TA, and ID are connected via wireless networks [37]. In wire networks, the communication bandwidth between the two machines is 20 MB/s, and the average network delay is 1 ms. In wireless networks, the communication bandwidth between the two machines is 4 MB/s, and the average network delay is 4 ms.

TABLE 2: Comparison with other related works.

Schemes	Data integrity	Supporting tolerance	Dynamic	Select outsource
Li et al.'s scheme [36]	✓	✓	✗	✗
Shen et al.'s scheme [18]	✓	✓	✓	✗
Fan et al.'s scheme [23]	✓	✗	✗	✗
TSVA-CP-ABE	✓	✓	✓	✓

TABLE 3: Computation cost comparisons.

Schemes	KP-TSABE [18]	VO-ABE [23]	TSVA-CP-ABE
Encryption	$\psi + (2 + S)\omega + \nu$	$\psi + \omega + (T + 2 + S)\nu$	$\psi + \omega + (T + 2 + S)\nu$
KeyGen	$ S (T + 6)\omega$	$\psi + S (T + 6)\omega$	$\psi + Y \omega + (n + 1)\nu$
Transform	$\emptyset$	$O(\text{tree}) + [4\nu Y + (T - n)]\omega + \nu$	$O(\text{tree}) + Y \omega + \nu$
Decryption	$O(\text{tree}) + [4\nu Y + (4 T + 11 - n)]\omega$	$\psi + n\nu$	$\psi + n\nu$

FIGURE 3: Cost of outsourced decryption.

FIGURE 4: Cost of decryption.

When considering data aggregation, our scheme is to verify the data and join the data that passes the verification. A hash function is used during verification, and the input parameters are ciphertext, timestamp, and secret key. Among them, the ciphertext is 512 bits, the secret key is 128 bits, the timestamp is 64 bits, and the hash function is 32 bits. The length of the data uploaded

by the device is $(512 + 64 + 32)$ bits (the secret key has been transmitted before). ECN only needs to perform one hash operation and two comparisons (one-time comparison and the other hash result comparison) to verify the legality of the data. The length of data uploaded by ECN is $(512n + 64 + 32)$ bits, where n is the number of IDs. MCP then verifies the uploaded data again. After passing

FIGURE 5: Cost of device.

FIGURE 6: Cost of edge node.

FIGURE 7: Cost of cloud.

the verification, an AES decryption can recover all the plaintext.

Our goal is to evaluate the efficiency of our scheme. Figure 3 shows the time of outsourcing, which also increases with the number of attributes. Figure 4 shows the time of

final decryption. Because most of the computational cost is in the outsourced phase, and the final decryption phase only needs to perform the exponential operation once. Therefore, the time of the final decryption is independent of the number of attributes. In particular, because IDs only need to perform

final decryption, this makes our system suitable for applications with limited resources.

When considering data aggregation, we separately calculate the time consumption of IDs, ECNs, and MCP. Here, we stipulate that the size of each uploaded data is 512 bits. For ID, only one AES encryption and one hash operation need to be performed each time. Therefore, the calculation cost will not increase with the number of devices. The computational cost is shown in Figure 5. The communication cost is the sum of the ciphertext's size, the length of the timestamp, and the length of the hash result, which is 608 bits.

For ECNs, only $2n$ comparisons and one hash operation are required each time; therefore, the computational cost will increase as the number of IDs increases. The computational cost is shown in Figure 6. The communication cost is the sum of the size of the aggregated ciphertext, the length of the timestamp, and the length of the hash result, which is $(512n + 64 + 32)$ bits. When the number of IDs increases to a certain extent, the number of ECNs can be appropriately increased to reduce the computational overhead of each ECN.

For MCP, only $2n$ comparisons, one AES decryption, and one hash operation are required each time. Therefore, the computational cost will increase as the number of IDs increases. The computational cost is shown in Figure 7. Most of the encryption and decryption calculations are assigned to ECN, which allows MCP to perform other tasks.

7. Conclusion

In this study, we proposed the TSVA-CP-ABE scheme for mobile crowdsensing. The proposed scheme supports key distribution and efficient data aggregation. ECN can assist ID to quickly obtain the session key and reduce the computing overhead of ID. At the same time, ECN can filter out the expired key and illegal aggregated data to save bandwidth. Combined with the revocation of attribute-based encryption, we have realized the dynamic joining and exiting of IDs. Performance analysis shows that, compared with traditional methods, our scheme can reduce computing overhead and communication costs and is very suitable for edge-assisted mobile crowdsensing. In our scheme, the verification of the outsourced results is verified by ID itself, and no additional fully trusted third party is required to verify. Our proposed system is the first TSVA-CP-ABE system that supports time-sensitive, revocation, and verifiable data. However, our system only supports detecting false data in plaintext. Therefore, our system is not suitable for the environment with a large number of malicious devices: malicious devices upload a large number of false data, and the platform can only be found after decryption, resulting in waste of computing resources. Our future work is to obtain a TSVA-CP-ABE system that can detect false data in the ciphertext.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (Grant nos. 2018YFE0207600 and 2018YFB2100403), the National Natural Science Foundation of Shaanxi Province (Grant nos. 2019ZDLGY13-03-01, 2020CGXNG-002, and 2019JQ-273), the Natural Science Foundation of China (Grant no. 61802243), and the Fundamental Research Funds for the Central Universities (Grant no. JB180305).

References

- [1] S. Guo and M.-O. Pun, "Indoor semantic-rich link-node model construction using crowdsourced trajectories from smartphones," *IEEE Sensors Journal*, vol. 19, no. 22, pp. 10917–10934, 2019.
- [2] L. Hu and Q. Ni, "IoT-driven automated object detection algorithm for urban surveillance systems in smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 747–754, 2018.
- [3] M. Z. Hasan and H. Al-Rizzo, "Optimization of sensor deployment for industrial Internet of Things using a multiswarm algorithm," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10344–10362, 2019.
- [4] Y. Shen, T. Zhang, Y. Wang, H. Wang, and X. Jiang, "MicroThings: a generic IoT architecture for flexible data aggregation and scalable service cooperation," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 86–93, 2017.
- [5] Y. Yang, N. Xiong, N. Y. Chong, and X. Défago, "A decentralized and adaptive flocking algorithm for autonomous mobile robots," in *Proceedings of the GPC Workshops*, pp. 262–268, IEEE Computer Society, Kunming, China, May 2008.
- [6] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge computing enabled smart cities: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, p. 10200, 2020.
- [7] T. Zhang, J. Ma, Q. Li, N. Xi, and C. Sun, "Trust-based service composition in multi-domain environments under time constraint," *Science China Information Sciences*, vol. 57, no. 9, pp. 1–16, 2014.
- [8] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.
- [9] T. Wang, G. Zhang, A. Liu, M. Z. A. Bhuiyan, and Q. Jin, "A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4831–4843, 2019.
- [10] K. Cao, L. Li, Y. Cui, T. Wei, and S. Hu, "Exploring placement of heterogeneous edge servers for response time minimization in mobile edge-cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 494–503, 2021.
- [11] T. Li, K. Ota, T. Wang, X. Li, Z. Cai, and A. Liu, "Optimizing the coverage via the UAVs with lower costs for information-centric Internet of Things," *IEEE Access*, vol. 7, pp. 15292–15309, 2019.
- [12] J. Xiong, R. Bi, M. Zhao, J. Guo, and Q. Yang, "Edge-assisted privacy-preserving raw data sharing framework for connected

- autonomous vehicles,” *IEEE Wireless Communications*, vol. 27, no. 3, pp. 24–30, 2020.
- [13] Q. Zhang, C. Zhou, N. Xiong, Y. Qin, X. Li, and S. Huang, “Multimodel-based incident prediction and risk assessment in dynamic cybersecurity protection for industrial control systems,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 10, pp. 1429–1444, 2016.
 - [14] Q. Chen, S. Tang, Q. Yang, and S. Fu, “Cooper: cooperative perception for connected autonomous vehicles based on 3D point clouds,” in *Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 514–524, Dallas, TX, USA, July 2019.
 - [15] R. Lu, X. Lin, H. Zhu, P. Ho, and X. Shen, “ECPP: efficient conditional privacy preservation protocol for secure vehicular communications,” in *Proceedings of the IEEE INFOCOM 2008—the 27th Conference on Computer Communications*, pp. 1229–1237, Phoenix, AZ, USA, April 2008.
 - [16] A. Shahzad, M. Lee, Y.-K. Lee et al., “Real time MODBUS transmissions and cryptography security designs and enhancements of protocol sensitive information,” *Symmetry*, vol. 7, no. 3, pp. 1176–1210, 2015.
 - [17] M. Huang, W. Liu, T. Wang, H. Song, X. Li, and A. Liu, “A queuing delay utilization scheme for on-path service aggregation in services-oriented computing networks,” *IEEE Access*, vol. 7, pp. 23816–23833, 2019.
 - [18] X. Shen, L. Zhu, C. Xu, K. Sharif, and R. Lu, “A privacy-preserving data aggregation scheme for dynamic groups in fog computing,” *Information Sciences*, vol. 514, pp. 118–130, 2020.
 - [19] S. Hu, L. Liu, L. Fang, F. Zhou, and R. Ye, “A novel energy-efficient and privacy-preserving data aggregation for WSNs,” *IEEE Access*, vol. 8, pp. 802–813, 2020.
 - [20] K. Huang, Q. Zhang, C. Zhou, N. Xiong, and Y. Qin, “An efficient intrusion detection approach for visual sensor networks based on traffic pattern learning,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 10, pp. 2704–2713, 2017.
 - [21] R. Zhang, J. Shi, Y. Zhang, and C. Zhang, “Verifiable privacy-preserving aggregation in people-centric urban sensing systems,” *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 268–278, 2013.
 - [22] Q. Li, G. Cao, and T. F. L. Porta, “Efficient and privacy-aware data aggregation in mobile sensing,” *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 2, pp. 115–129, 2014.
 - [23] C.-I. Fan, S.-Y. Huang, and Y.-L. Lai, “Privacy-enhanced data aggregation scheme against internal attackers in smart grid,” *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 666–675, 2014.
 - [24] J. Won, C. Y. T. Ma, D. K. Y. Yau, and N. S. V. Rao, “Proactive fault-tolerant aggregation protocol for privacy-assured smart metering,” in *Proceedings of the IEEE INFOCOM 2014—IEEE Conference on Computer Communications*, pp. 2804–2812, Toronto, ON, Canada, April 2014.
 - [25] H. Wang, Z. Wang, and J. Domingo-Ferrer, “Anonymous and secure aggregation scheme in fog-based public cloud computing,” *Future Generation Computer Systems*, vol. 78, no. 2, pp. 712–719, 2018.
 - [26] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, “A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT,” *IEEE Access*, vol. 5, pp. 3302–3312, 2017.
 - [27] K.-A. Shim and C.-M. Park, “A secure data aggregation scheme based on appropriate cryptographic primitives in heterogeneous wireless sensor networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 8, pp. 2128–2139, 2015.
 - [28] L. Shen, J. Ma, X. Liu, F. Wei, and M. Miao, “A secure and efficient ID-based aggregate signature scheme for wireless sensor networks,” *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 546–554, 2017.
 - [29] H. Bao and R. Lu, “A lightweight data aggregation scheme achieving privacy preservation and data integrity with differential privacy and fault tolerance,” *Peer-to-Peer Networking and Applications*, vol. 10, no. 1, pp. 106–121, 2017.
 - [30] A. Beimel, “Secure schemes for secret sharing and key distribution,” Ph.D. thesis, Technion-Israel Institute of Technology, Haifa, Israel, 1996.
 - [31] B. Qin, R. H. Deng, S. Liu, and S. Ma, “Attribute-based encryption with efficient verifiable outsourced decryption,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 7, pp. 1384–1393, 2015.
 - [32] S. Yu, C. Wang, K. Ren, and W. Lou, “Achieving secure, scalable, and fine-grained data access control in cloud computing,” in *Proceedings of the 2010 IEEE INFOCOM*, pp. 1–9, IEEE, San Diego, CA, USA, March 2010.
 - [33] M. Green, S. Hohenberger, and B. Waters, “Outsourcing the decryption of ABE ciphertexts,” in *Proceedings of the 20th USENIX Conference on Security*, p. 34, USENIX Association, San Francisco, CA, USA, August 2011.
 - [34] B. Qin and D. Zheng, “Generic approach to outsource the decryption of attribute-based encryption in cloud computing,” *IEEE Access*, vol. 7, pp. 42331–42342, 2019.
 - [35] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pp. 89–98, Alexandria, VA, USA, October 2006.
 - [36] X. Li, S. Liu, F. Wu, S. Kumari, and J. J. P. C. Rodrigues, “Privacy preserving data aggregation scheme for mobile edge computing assisted IoT applications,” *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4755–4763, 2019.
 - [37] W. Wu, N. Xiong, and C. Wu, “Improved clustering algorithm based on energy consumption in wireless sensor networks,” *IET Networks*, vol. 6, no. 3, pp. 47–53, 2017.