Research Article

Leveraging Mobile Nodes for Preserving Node Privacy in Mobile Crowd Sensing

Qinghua Chen ^{1,2}, **Shengbao Zheng**,³ and **Zhengqiu Weng** ²

¹College of Computer Science & Technology, Zhejiang University of Technology, Hangzhou, China

²Department of Information Technology, Wenzhou Vocational & Technical College, Wenzhou, China

³Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

Correspondence should be addressed to Qinghua Chen; kegully@qq.com

Received 29 December 2017; Accepted 25 March 2018; Published 30 April 2018

Academic Editor: Lu Wang

Copyright © 2018 Qinghua Chen 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 crowd sensing has been a very important paradigm for collecting sensing data from a large number of mobile nodes dispersed over a wide area. Although it provides a powerful means for sensing data collection, mobile nodes are subject to privacy leakage risks since the sensing data from a mobile node may contain sensitive information about the sensor node such as physical locations. Therefore, it is essential for mobile crowd sensing to have a privacy preserving scheme to protect the privacy of mobile nodes. A number of approaches have been proposed for preserving node privacy in mobile crowd sensing. Many of the existing approaches manipulate the sensing data so that attackers could not obtain the privacy-sensitive data. The main drawback of these approaches is that the manipulated data have a lower utility in real-world applications. In this paper, we propose an approach called P^3 to preserve the privacy of the mobile nodes in a mobile crowd sensing system, leveraging node mobility. In essence, a mobile node determines a routing path that consists of a sequence of intermediate mobile nodes and then forwards the sensing data along the routing path. By using asymmetric encryptions, it is ensured that a malicious node is not able to determine the source nodes by tracing back along the path. With our approach, upper-layer applications are able to access the original sensing data from mobile nodes, while the privacy of the mobile node is not compromised. Our theoretical analysis shows that the proposed approach achieves a high level of privacy preserving capability. The simulation results also show that the proposed approach incurs only modest overhead.

1. Introduction

Mobile crowd sensing has been a very important paradigm for collecting sensing data from a large number of mobile nodes dispersed over a wide area. A mobile crowd sensing system [1] has the following basic framework of data collection. First, smartphone nodes collect environmental data with the embedded sensors, such as noise level, air pollution level, GPS trajectories [2, 3], and radio signal strength [4–6]. Next, these nodes send the data directly to the central server (note that the server is able to get the ID of the data sender). Finally, the server uses all the collected data to do some computations and analysis and finally draw some conclusions for further use. A wide spectrum of applications of mobile crowd sensing is envisioned, such as road traffic sensing [7], traffic light sensing [8], urban monitoring [9], and indoor localization [10, 11].

Although it provides a powerful means for sensing data collection, mobile nodes are subject to privacy leakage risks since the sensing data from a mobile node may contain sensitive information about the sensor node such as physical locations [12]. For instance, one intends to collect the noise level of a city and proposes a task. The sensing data contain not only the noise information, but also the location information acquired by GPS. Therefore, the data server easily matches the participant ID and the user's location. If the sensing is continuous, the time-series data may even reveal the trajectories of participants. This can be used to locate the sensitive locations (e.g., home, workplace) of participants. A smartphone user may be willing to participate in sensing but may not want to disclose his location. Thus, it is of great importance to preserve privacy throughout the data collection process in mobile crowd sensing.

Therefore, it is essential for mobile crowd sensing to have a privacy preserving scheme to protect the privacy of mobile nodes. It is challenging for mobile crowd sensing to preserve node privacy while maintaining a high level of the utility of the collected sensing data. A number of approaches [13, 14] have been proposed for preserving node privacy in mobile crowd sensing. Many of the existing approaches manipulate the sensing data so that attackers could not obtain the privacy-sensitive data. The main drawback of these approaches is that the manipulated data have a lower utility in real-world applications.

In this paper, we propose an approach called P^3 to preserve the privacy of the mobile nodes in a mobile crowd sensing system. In essence, a mobile node determines a routing path that consists of a sequence of intermediate mobile nodes and then forwards the sensing data along the routing path. By using asymmetric encryptions, it is ensured that a malicious node is not able to determine the source nodes by tracing back along the path. Note that our approach leverages mobile nodes to protect the privacy of a node generating sensing data. In addition, mobile nodes are dynamic and may come and leave the system at any time. Routing paths consist of different sets of mobile nodes. With our approach, upper-layer applications are able to access the original sensing data from mobile nodes, while the privacy of the mobile node is not compromised. Our theoretical analysis shows that the proposed approach achieves a high level of privacy preserving capability. The simulation results also show that the proposed approach incurs only modest overhead.

The rest of the paper is structured as follows. Section 2 reviews the related work. Section 3 presents the problem model and formally defines the problem. Section 4 presents the proposed approach of P^3 . Section 5 provides the theoretical analysis of our approach. The performance evaluation is presented in Section 6. Finally, we conclude our paper in Section 7.

2. System Model and Problem Formulation

2.1. System Model. In this paper, our system is composed of three parts—smartphones, a central server, and a public key handler. Each smartphone is regarded as a node in our system, and each one knows its own data. Each node also has a public key and a private key for encryption and decryption. Just as its name, the public key can be seen by every other node, and all the public keys are stored in the public key handler. However, on the contrary, the private key is only kept by the node itself. No one can know it except itself. The central server intends to collect all the sensing data provided by nodes and utilizes them to form knowledge.

Now assume that there are N online nodes in our system. The nodes are denoted as u_i ($i = 1, 2, 3, \dots, N$). Besides, assume each node has M sensing data; therefore, the sensing data are denoted as

$$D_i = \{d_{i1}, d_{i2}, d_{i3}, \dots, d_{ij}\} \quad (1)$$

($i = 1, 2, 3, \dots, N$; $j = 1, 2, 3, \dots, M$), where d_{ij} means the j th data of node u_i . In particular, when each node has only one

data, the data set of node u_i has only one parameter d_{i1} , and we simplify it to D_i .

Furthermore, the data collection function of the server S is denoted as f . Obviously,

$$f = \bigcup_{i=1}^N \bigcup_{j=1}^M d_{ij}. \quad (2)$$

In particular, when each node has only one sensing data, it is simplified to

$$f = \bigcup_{i=1}^N D_i. \quad (3)$$

We define a collection set of sensing data

$$C = \{D_1, D_2, D_3, \dots, D_i\}. \quad (4)$$

Besides, for each node, it has a public key and a private key. Take node u_i as an example; its public key is denoted as P_{K_i} and its private key is denoted as S_{K_i} . All the public keys are stored at the public key handler, H , and it has a set of public keys

$$S = \{P_{K_1}, P_{K_2}, P_{K_3}, \dots, P_{K_i}\}. \quad (5)$$

There also exists a global clock in our system, and every node synchronizes their clocks to it to get the current time t . The key handler also calculates a forwarding threshold T for each node. Each time a node attempts to forward a data, it needs to check whether the current time t is before T ($t < T$). If not, drop the data.

At last, we use a package to represent the data transferred on the forwarding routes, which is denoted as P , such as $P_1, P_2, \dots, P_m$.

The definitions of all notations adopted are described in “Notations.”

2.2. Design Requirements. Obviously, the most significant aspect of our privacy preserving data collection approach is protecting the privacy. But there also exist other aspects such as the efficiency of data forwarding that we need to consider. The following two aspects are the main requirements of our privacy preserving data collection approach.

(1) *Preserving Node Privacy.* We treat each smartphone as a node in our system, either honest or malicious. The attacker in our system can either be the server or any other node in the mobile crowd sensing network. The target of our aim is to realize anonymity of sensing data and make sure the server cannot match the sensing data with their originators. In other words, our goal is that the server could only get the sensing data while not knowing whom this data belongs to.

(2) *Maintaining High Efficiency.* The mobile crowd sensing system is usually a real-time system, in which the real-time attribute of sensing data is significant. In addition, due to the limitation of the smartphones, the instability of battery duration, and the availability of smartphones, the efficiency

of our data collection approach must be taken into account. The processing ability was also a limitation in the past, but as the development of smartphones, it becomes not as important as before. In this paper, we use the overhead to evaluate the efficiency, including the time cost of encryption, decryption, data forwarding, and storage utilization.

2.3. The Attack Model. In our system, we have three different types of entities: a server, a public key handler, and N online smartphone nodes. First, we assume the public key handler is fully trusted. Second, we assume that there are malicious nodes that collude with the server and the number is up to γN ($0 < \gamma < 1$). We also assume honest nodes can keep their private keys securely. Each node sees the package addressed to it but cannot see packages between other nodes. Honest nodes will forward the new package at the time point as instructed in the old package. The goal of the adversary is to identify the originator of the package from the given piece of data. We suppose that the attacker is successful if he/she determines the node that generated the data, out of all the nodes in the network.

3. Design of P^3

In this section, we present two parts: the first part is an overview of our privacy preserving data collection approach, and we describe our approach in detail in the second part. The approach consists of four phases: setup, route selection, data encryption, and data forwarding.

Figure 1 shows the system architecture of our approach. The proposed system consists of a central server, a public key handler, and some smartphone devices. The central server aims to collect all the sensing data. And the public key handler is responsible for managing all the public keys.

In our system, the public key handler has an additional function—checking the state of a node. It says *Hello* to each node periodically. If a node is online, then its name will be on the online list. However, on the contrary, if a node is offline, its name will be removed from the list. This guarantees that there will not be an offline node on the forwarding route which leads to package missing. Besides, the selection of forwarding routes is also conducted by the public key handler.

Each node obtains sensing data through the sensors embedded in their smartphones. It then sends a sending query to the public key handler and encrypts the original data into a package with the public keys provided by the public key handler and then sends this package to the next node.

Once the next node receives this data, it decrypts the package with its own private key. Then it will get three parts from it—the address of the next node, the time threshold, and the package to be sent. After checking the time requirement, the next step is to pass the package to the next node and iterate this until the central server gets the data. What the server needs to do is to add this data into the data collection set.

The first step of our approach is the system setup. In a crowd sensing system, nodes may come and go. In this step, the central server first needs to confirm the nodes and then inform them the sensing task. Besides, each node should get prepared for sensing and initialize their sensors. In order to

encrypt the data, each node should build a pair of key—a public key used for encrypting the data and a private key for decrypting the package. This pairs of keys are constant and will not change. The private key is stored by the node locally, while the public key is stored by a fully trusted third party—the public key handler.

The public key handler acquires the name list of nodes from the central server and contact with them. It triggers a query by a *Hello* message to each node to make sure whether they are online or not. The first time a node receives the query, it passes its public key to the public key handler to answer *Hello*. After all the available nodes pass their public keys to the public key handler, it then forms a public key set of all the nodes. Meanwhile, an available nodes list is built as well. Besides, due to the limitation of smartphones, some nodes may be unavailable at some time. So we assume the public key handler is also in charge of checking the state of the nodes. It may send a *Hello* message to all the nodes. As soon as a node receives the message, it will answer with a signal. If the public key handler does not receive this signal, this node will be considered to be unavailable and cannot be utilized for forwarding package. Then it will be removed from the available nodes list. By this way, the public key handler checks whether a smartphone node is online.

This process is carried out periodically. The period will be neither too long nor too short. If it is set too long, some nodes may have been offline but not removed from the available nodes list, which will lead to some forwarding routes' breaking. On the other hand, if set too short, the communication overhead may be too large. Dynamics does not create a problem to our protocol. The data transmission finishes in a very short time. In case a transmission is stopped by a node going offline, the data is just ignored. In a crowd sensing system, data are typically redundant and small loss of data is tolerable.

3.1. Determining a Data Forwarding Path. The second step is to select a suitable route to forward the sensing data. All the mobile phones are addressed by IP addresses. As we have assumed before, there exists an available nodes list in the key handler, which is fully trusted. So it can generate a forwarding route for each node and send the routes to them. In this way, each data originator can easily know who can be depended on to upload its data. During the system setup step, we need to set a hop number n , which means how many nodes participate in this forwarding process. However, we can also rule a range for n , which has an upper bound and a lower bound, to avoid attackers using the constant number of hops to guess the data originator. To simplify the analysis, we only discuss the constant n . On the forwarding route, except the beginning node, there is also the need for $n - 1$ extra nodes. As we have set in the model part, there are N online nodes in the system, so this route selection problem is equivalent to a combination problem, that is, to select $n - 1$ nodes from $N - 1$ nodes, so the number of combination methods is

$$C_{N-1}^{n-1} = \frac{(N-1)!}{(n-1)!(N-n)!}. \quad (6)$$

FIGURE 1: The main framework of the proposed approach.

It also means the number of routes can be selected when a sensing data originator intends to pass a data to the server. What to do next is just to select one route from them. The route information that public key handler sends to a node consists of the intermediate nodes' public key and their address.

Note that we assume that the data communications between any pair of nodes are always possible through the Internet. It is practical in the real world since any mobile node has the cellular Internet access.

3.2. Encrypting along the Forwarding Path. We use an example to illustrate how to encrypt the data. Figure 2 shows the process of data encryption. Node A intends to send its sensing data d_A to the central server and the intermediate nodes are node B and node C. P_1 represents the package that node A sends to B and P_2 represents the package that node B sends to node C. P_3 is what C sends to the central server.

First, node A issues a query to the public key handler for a route to upload data and the public key handler answers the query with a forwarding route information, which contains the public keys of nodes B, C, the central server, and their address, as well as some time thresholds. When the node A receives the route, it then encrypts the sensing data d_A with the server's public key P_{K_S} and gets P_3 . After that it needs to combine three parts: P_3 , the route information, which is the server's address, and the time threshold used to judge whether to send the package out or drop it. After it gets the combination, encrypt this with node C's public key. This is P_2 . Next, utilize the node B's public key to encrypt the combination of P_2 , the route information, node C's address, and the time threshold and generate P_1 . This is what node A sends to node B. In addition, node A can also get node B's address and a time threshold.

Besides, we also utilize methods to ensure all the packages have the same size, so that attackers cannot judge the number of nodes on the route by means of analyzing the size changes of the packages.

3.3. Forwarding Sensing Data along the Path. The final step of our privacy preserving data collection approach is the data forwarding, which describes how to move the encrypted packages. With the P_s we have got, what we need to do is to transfer them along the route.

First, node A sends P_1 to node B at the time threshold T_1 . Once node B gets P_1 , decrypt it with its private key S_{K_B} and get P_2 . Besides, it is also aware of the destination, node C's address. However, it needs to get the current time t and compare it with the time threshold T_2 . If t is bigger than T_2 , it means that this package is out of date and should be dropped. If not dropped, the package will arrive at the server very late and the server can easily identify its originator. If the time requirement is satisfied, send P_2 to node C at the time threshold T_2 . As soon as P_2 arrives, node C decrypts it with S_{K_C} and verifies the time requirement and, if satisfied, issues it to the next hop, the central server, at the time threshold T_3 . The central server can easily acquire the exact sensing data d_A by decrypting the P_3 it received but without acknowledging its originator.

All the nodes use this method to hand their sensing data to the central server. In a short time, the server will receive all the data and build a data collection set.

4. Theoretical Analysis of P^3

The most essential aspect to evaluate a privacy preserving problem is its security level. In this section, we provide a theoretical security analysis of our privacy preserving data collection approach.

In the paper, we assume the server is unauthentic and has the authority to get all the sensing data but should not know their originators, though it is eager to. So privacy is defined as the relationship or the ownership between sensing data and their originators, which are nodes in our system. Our target is to realize the anonymity of the sensing data. For this kind of privacy, the server is the attacker. It cannot distinguish to

FIGURE 2: Data encryption.

whom each data belongs in our system. Besides, there may be other attackers who are nodes. They collude with the central server to help it realize the attack target.

The malicious nodes on the forwarding route can infer the originator of the data by means of colluding with each other. The server can query its former node that it received data from. A malicious node will answer the query, but an honest node would not answer this query. The basic idea is that the server asks its former node who sent the package to it. If this node answers the query, it would ask its former node the same question, until it gets no answer. And we can assume that the node which does not reply to the query is the beginning node (the originator of the sensing data). When all the nodes on the forwarding route except the data originator are malicious, the attack target is achieved.

However, this is just the simplest way to finish an attack, and there exist some clever means as well. For example, if there exists an honest node between two malicious nodes, as Figure 3 shows, where nodes in dark color represent malicious nodes and those in light color are honest (Figures 4 and 5 use the same way to indicate a node). When node *D* queried, it will say that node *C* sent data to it. And when node *C* queried, it will not reply to the query. So the attack appears to have failed. However, node *B* may tell the server that it has sent data to node *C* and it receives it from node *A*. So the server may infer that there may be a route $A \rightarrow B \rightarrow C \rightarrow D$.

The security level of our approach depends on the distribution of malicious nodes in the route. We can classify all possible distributions into three cases:

- (i) Case 1: the next hop of the originator is honest.
- (ii) Case 2.1: the next hop of the originator is malicious, and there are two consecutive nodes on the route which are honest.
- (iii) Case 2.2: the next hop of the originator is malicious, and there are not two consecutive nodes on the route which are honest.

4.1. Case 1. The next hop of the originator is honest.

In this case, as is showed in Figure 4, we know that if the next hop of the data originator on the forwarding route is an honest node, it will not answer any query from the server. So the originator's identity will not be exposed. Therefore, the server will never figure out the originator of the sensing data transferred by this route. The best circumstance is that the server can infer that the second node on the route is the owner of the data. Actually, it is just an intermediate node that is honest, but not the data originator.

The probability of this case is $1 - \gamma$, which equals the probability of the next node of the data originator on the forwarding route which is honest.

Theorem 1. In Case 1, the success probability of the attacker is $1/(1 - \gamma)N$.

Proof. When an attacker receives a piece of data, it will attempt to link the data to the originator by tracing the data back. For each node, it knows only who are the senders or receivers of the packages it receives or sends. In Case 1, we know that if the next hop of the originator is honest; thus, it will not disclose who is the sender of a package to the attackers. Therefore, even all other nodes in the route are malicious; the server cannot do better than a random guess. Since the data originator can be any honest node, the success probability of attackers is $1/(1 - \gamma)N$. $\square$

When analyzing this case, we know that it has no relationship with the number of hops n . The total probability of attackers guessing correctly under Case 1 is

$$P = \frac{P_{\text{case 1}}}{(1 - \gamma)N} = \frac{1}{N}. \quad (7)$$

Table 1 shows the concrete values of p .

4.2. Case 2. If the next hop of the originator is a malicious node. This case is divided into two cases: there exist two consecutive honest nodes on the forwarding route and two consecutive honest nodes do not exist. The latter case we will discuss is Case 2.2.

4.3. Case 2.1. The next hop of the data originator is a malicious node and there exist two consecutive honest nodes on the forwarding route.

If there exist two consecutive honest nodes on the forwarding route, the server cannot finish an attack, because the server cannot restore the route. For example, Figure 5 shows this circumstance. Node *B* tells the server that node *A* sent a package to it and it has forwarded it to node *C*. However, nodes *C* and *D* are honest nodes and will not answer any question of the server and malicious nodes. So although the next hop of node *D*, which is *E*, is malicious, it can only provide the information that it received a package from *D*. With only this information, the server cannot restore the whole route because of the absence of relation between *C* and *D*. The server may consider that these nodes belong to two routes. So the privacy is preserved.

However, the malicious nodes may add a large delay before they forward package. As a result, an honest node's

FIGURE 3: Clever guess.

FIGURE 4: Case 1.

FIGURE 5: Case 2.1.

sensing data may arrive at the server very late. The first malicious node can tell the server who sent this late-arrived data, and the privacy we wish to preserve is exposed. We know that the trusted public key handler also assists in selecting the forwarding routes. Besides, the encryption time of data originator can be estimated; meanwhile, the forwarding time and the decryption time of each node can also be estimated. So the public key handler can calculate the package's arrive time for each node. To avoid the estimating deviation resulting in dropping package by mistake, a small value is added to release the strict limitation of time threshold. So when the key handler selects a route, it also calculates the time threshold for each node. And with it, the malicious delay cannot help the attack.

The probability of Case 2.1 is

$$p_{\text{case 2.1}} = \gamma - \sum_{i=0}^{\lfloor (n-1)/2 \rfloor} \frac{P_{\gamma N}^{(n-1-i)} P_{(n-1-i)}^i C_{(1-\gamma)N-1}^i}{P_{N-1}^{n-1}} \quad (8)$$

Theorem 2. In Case 2.1, the success probability of attackers is $1/(1-\gamma)N$.

Proof. In this case, the next hop of the data originator is malicious; thus, it knows who is the originator. However, the data package is encrypted at this hop and the cipher-text cannot be matched to the data received by the server. Therefore, to link a piece of data to its originator, the attackers still need to trace back. If there are two consecutive honest nodes in the route, tracing back will fail at this point. Then similar to Case 1, the success probability of attackers is $1/(1-\gamma)N$. $\square$

Therefore, the total probability p is

$$p = \frac{p_{\text{case 2.1}}}{(1-\gamma)N}. \quad (9)$$

Tables 2, 3, and 4 show the exact values of this circumstance. From Tables 2, 3, and 4, we draw the conclusion that when the number of hops n is larger than 10, n has little impact on p . What influence it are γ and N .

4.4. Case 2.2. The next hop of the data originator is a malicious node and two consecutive honest nodes on the forwarding route do not exist.

In this case, because the second hop node is malicious, the server can acknowledge that this node sent the data. So what it needs to do is to identify the data transferred by this route. However, if no consecutive two honest nodes on the route exist, it is not hard to know this data. The simplest circumstance is that all the nodes after the second node are malicious. The server just needs to ask the former node until a node does not answer the query. While the most complex circumstance is that there is an honest node between every two malicious nodes, as Figure 3 shows. But the server and malicious nodes can still complete the attack. With the clever guess means we talked above, we know that a Malicious-Honest-Malicious combination can also be restored, though an honest node exists. Therefore, this case can be completely attacked.

The probability of Case 2.2 is

$$p_{\text{case 2.2}} = \sum_{i=0}^{\lfloor (n-1)/2 \rfloor} \frac{P_{\gamma N}^{(n-1-i)} P_{(n-1-i)}^i C_{(1-\gamma)N-1}^i}{P_{N-1}^{n-1}}. \quad (10)$$

Theorem 3. In Case 2.2, the success probability of attackers is 1.

Proof. In this case, the simplest circumstance is that all the nodes after the second node are malicious. While the most complex circumstance is that there is an honest node between every two malicious nodes. With the basic and the clever guess means we talked above, the package forwarding route can also be restored, which means a successful attack. So the success probability of attackers is 1. $\square$

Tables 5, 6, and 7 show that the probability of Case 2.2 happens when γ is 0.03, 0.05, and 0.1.

5. Performance Evaluation

Our simulation experiment simulates the whole process of conducting a sensing task, including the central server allocating sensing task, the public key handler collecting public keys, and selecting random forwarding route for each node, all the nodes sending out their sensing data and the server colluding with the malicious nodes to realize attack. By

TABLE 1: Probability of attackers guessing correctly under Case 1.

	$\gamma = 0.03$	$\gamma = 0.05$	$\gamma = 0.10$
$N = 4000$	$2^{-11.97}$	$2^{-11.97}$	$2^{-11.97}$
$N = 6000$	$2^{-12.55}$	$2^{-12.55}$	$2^{-12.55}$
$N = 8000$	$2^{-12.97}$	$2^{-12.97}$	$2^{-12.97}$
$N = 10000$	$2^{-13.29}$	$2^{-13.29}$	$2^{-13.29}$

TABLE 2: Probability of attackers guessing correctly under Case 2.1 when $\gamma = 0.03$.

$N = 4000$	n	11	12	13	14	15
$\gamma N = 120$	p	$2^{-16.98}$	$2^{-16.98}$	$2^{-16.98}$	$2^{-16.98}$	$2^{-16.98}$
$N = 6000$	n	11	12	13	14	15
$\gamma N = 180$	p	$2^{-17.57}$	$2^{-17.57}$	$2^{-17.57}$	$2^{-17.57}$	$2^{-17.57}$
$N = 8000$	n	11	12	13	14	15
$\gamma N = 240$	p	$2^{-17.98}$	$2^{-17.98}$	$2^{-17.98}$	$2^{-17.98}$	$2^{-17.98}$
$N = 10000$	n	11	12	13	14	15
$\gamma N = 300$	p	$2^{-18.30}$	$2^{-18.30}$	$2^{-18.30}$	$2^{-18.30}$	$2^{-18.30}$

TABLE 3: Probability of attackers guessing correctly under Case 2.1 when $\gamma = 0.05$.

$N = 4000$	n	11	12	13	14	15
$\gamma N = 120$	p	$2^{-16.21}$	$2^{-16.21}$	$2^{-16.21}$	$2^{-16.21}$	$2^{-16.21}$
$N = 6000$	n	11	12	13	14	15
$\gamma N = 180$	p	$2^{-16.80}$	$2^{-16.80}$	$2^{-16.80}$	$2^{-16.80}$	$2^{-16.80}$
$N = 8000$	n	11	12	13	14	15
$\gamma N = 240$	p	$2^{-17.21}$	$2^{-17.21}$	$2^{-17.21}$	$2^{-17.21}$	$2^{-17.21}$
$N = 10000$	n	11	12	13	14	15
$\gamma N = 300$	p	$2^{-17.54}$	$2^{-17.54}$	$2^{-17.54}$	$2^{-17.54}$	$2^{-17.54}$

TABLE 4: Probability of attackers guessing correctly under Case 2.1 when $\gamma = 0.1$.

$N = 4000$	n	11	12	13	14	15
$\gamma N = 120$	p	$2^{-15.14}$	$2^{-15.14}$	$2^{-15.14}$	$2^{-15.14}$	$2^{-15.14}$
$N = 6000$	n	11	12	13	14	15
$\gamma N = 180$	p	$2^{-15.72}$	$2^{-15.72}$	$2^{-15.72}$	$2^{-15.72}$	$2^{-15.72}$
$N = 8000$	n	11	12	13	14	15
$\gamma N = 240$	p	$2^{-16.14}$	$2^{-16.14}$	$2^{-16.14}$	$2^{-16.14}$	$2^{-16.14}$
$N = 10000$	n	11	12	13	14	15
$\gamma N = 300$	p	$2^{-16.46}$	$2^{-16.46}$	$2^{-16.46}$	$2^{-16.46}$	$2^{-16.46}$

TABLE 5: Probability of attackers guessing correctly under Case 2.2 when $\gamma = 0.03$.

$N = 4000$	n	11	12	13	14	15
$\gamma N = 120$	p	$2^{-25.06}$	$2^{-27.92}$	$2^{-30.03}$	$2^{-32.78}$	$2^{-35.00}$
$N = 6000$	n	11	12	13	14	15
$\gamma N = 180$	p	$2^{-25.02}$	$2^{-27.85}$	$2^{-29.97}$	$2^{-32.70}$	$2^{-34.91}$
$N = 8000$	n	11	12	13	14	15
$\gamma N = 240$	p	$2^{-24.99}$	$2^{-27.83}$	$2^{-29.93}$	$2^{-32.65}$	$2^{-34.86}$
$N = 10000$	n	11	12	13	14	15
$\gamma N = 300$	p	$2^{-24.98}$	$2^{-27.81}$	$2^{-29.91}$	$2^{-32.63}$	$2^{-34.83}$

TABLE 6: Probability of attackers guessing correctly under Case 2.2 when $\gamma = 0.05$.

$N = 4000$	n	11	12	13	14	15
$\gamma N = 120$	p	$2^{-21.14}$	$2^{-23.40}$	$2^{-25.28}$	$2^{-27.48}$	$2^{-29.42}$
$N = 6000$	n	11	12	13	14	15
$\gamma N = 180$	p	$2^{-21.11}$	$2^{-23.37}$	$2^{-25.44}$	$2^{-27.43}$	$2^{-29.37}$
$N = 8000$	n	11	12	13	14	15
$\gamma N = 240$	p	$2^{-21.10}$	$2^{-23.35}$	$2^{-25.22}$	$2^{-27.40}$	$2^{-29.34}$
$N = 10000$	n	11	12	13	14	15
$\gamma N = 300$	p	$2^{-21.09}$	$2^{-23.34}$	$2^{-25.21}$	$2^{-27.39}$	$2^{-29.32}$

TABLE 7: Probability of attackers guessing correctly under Case 2.2 when $\gamma = 0.1$.

$N = 4000$	n	11	12	13	14	15
$\gamma N = 120$	p	$2^{-15.76}$	$2^{-17.33}$	$2^{-18.79}$	$2^{-20.33}$	$2^{-21.82}$
$N = 6000$	n	11	12	13	14	15
$\gamma N = 180$	p	$2^{-15.74}$	$2^{-17.31}$	$2^{-18.77}$	$2^{-20.31}$	$2^{-21.79}$
$N = 8000$	n	11	12	13	14	15
$\gamma N = 240$	p	$2^{-15.74}$	$2^{-17.30}$	$2^{-18.76}$	$2^{-20.30}$	$2^{-21.78}$
$N = 10000$	n	11	12	13	14	15
$\gamma N = 300$	p	$2^{-15.73}$	$2^{-17.30}$	$2^{-18.75}$	$2^{-20.29}$	$2^{-21.77}$

changing different parameters, we have acquired different results. Through analyzing them, we can draw the conclusion that our approach has a high security level.

Figure 6 shows the experimental ratio of exposed data originators of different γ . The number of all the online nodes N is 10000 and we care about the circumstance that γ equals 0.05 and 0.10. The range of hops is from 9 to 14. The simulation results of the experiment are similar to our theoretical analysis. And when then proportion of malicious nodes γ is 0.10 and the system select $n = 9$ as the number of hops, the ratio of exposed data originators is nearly 1.50×10^{-4} , which means that the server can only identify 1.50 sensing data's originator. This is clearly a beautiful result.

Figure 7 shows the different ratios of exposed data originators changing tendency as the number of hops changing in theoretical analysis. The number of online nodes N in this figure is still 10000. From the figure, we can see that as the number of hops increases, the ratio of exposed data originators declines and finally converges towards zero. All the ratios are in a magnitude of 10^{-4} . Obviously, when the number of hops is constant, the results show that the higher γ the system has, the less data originators are exposed.

The comparison of experimental and theoretical results is shown in Figure 8. The parameters are set as follows: $\gamma = 0.1$ and $N = 10000$. We can easily find that two lines which represent experimental and theoretical results nearly overlap. This is a strong evidence to prove our security analysis discussed before is right.

6. Related Work

Privacy preservation is a promising area in wireless networks [15], where there have been many applications [16, 17]. Many works have addressed varieties of privacy and security issues

in mobile crowd sensing networks, such as [18–23]; however, the goal of data collection is omitted. From the existing works, it is not hard to find some excellent works related to data aggregation, which have some aspects similar to our work and can be referred to, though having some differences with our work. Unfortunately, some of those existing works that solve the privacy preservation in data aggregation usually assume a trusted aggregator or server. They cannot deal with the circumstance that the aggregator or the server is unauthentic, for instance, [24–27].

Generally, data aggregation usually cares about the statistic results of the data, not the data themselves. Sometimes the real values of data are less useful than the statistic results. For example, we commonly prefer to acknowledge the maximum or the minimum or the average of a set of data. The real values of data are not necessary. The most common data aggregation is *Sum* and *Max/Min*: Yang et al. [13] proposed an encryption approach that helps an aggregator obtain the *Sum* of data without knowing each participants' specific data. Shi et al. [28, 29] proposed a construction for *Sum* aggregation which does not need the extra round of interaction. Li and Cao [14] proposed an efficient protocol to obtain the *Sum* aggregation, which employs an additive homomorphic encryption and a novel key management technique to support large plaintext space. And they also extend *Sum* aggregation to *Min* aggregation. Although these approaches solve the privacy preservation in data aggregation, they only provide *Sum* and *Min* function.

While, on the other hand, our work and data collection, which aims to collect all the data, must know all the exact values. Merely obtaining statistic results cannot satisfy our request, but these existing works also have significance to us. Boutsis and Kalogeraki [30] proposed an efficient way to collect location information based on exchanging data with other participants. But there is no central server in its model.

FIGURE 6: Experimental ratio of exposed data originators.

FIGURE 7: Theoretical ratio of exposed data originators.

FIGURE 8: Comparison of experimental and theoretical results.

Besides, the participants may be malicious, and it will lead to an unguaranteed security level.

The most frequently adopted method is slicing the data first and then mixing it. Shi et al. [31] utilized this method and proposed an aggregation approach that realized the privacy preservation. Destroying original data has little impact on

data aggregation, while for data collection, restoring the data is troubling. In order to mix the data slice, each slice coming from the same original data needs to be signed with the same ID, which is harmful to realizing the anonymity of data.

Data perturbation is another common technique. Many existing works protect individuals' privacy by adding noise to

the original data at the client side and allows the server to reconstruct the statistics of the original data [32–34]. Kargupta et al. [35] utilized the independent random noise, which can be separated from the perturbed data by studying the spectral properties of the perturbed data. Zhang et al. [36] proposed a data perturbation algorithm that can be used to protect participants' private information. But the disadvantage of these works is that the server can only acquire statistics result. Privacy preservation problem also exists in other networks, such as wireless sensor networks. Although our work focuses on the mobile crowd sensing network, there are many works in wireless sensor networks that can be referred to. The privacy preserving problem in wireless sensor networks usually has the same attack model as our work. The difference between these two networks is that wireless sensor networks usually have a constant topology, which seldom changes. However, in mobile crowd sensing networks, almost every participant moves all the time. Besides, some smartphones may be unavailable at some time. But if we only care about the method they preserve privacy, it is similar to our work.

Horney et al. [37] proposed a set of protocols that enable anonymous data collection in a sensor network. Sensor nodes, instead of transmitting their actual data, transmit a sample of the data complement to a base station, which then uses the negative samples to reconstruct a histogram of the original sensor readings. There also exist some approaches that can only do an approximate query. Besides, the topology graphs must be known to each node in most approaches in wireless sensor networks, which demands that each node owns enough storage and calculating ability. Therefore, these approaches are unpractical in mobile crowd sensing networks.

However, we propose a privacy preserving data collection approach. Onion routing approach [38] which operates by dynamically building anonymous connections within a network is referred. Specially, our approach has the following advantages over the previous approach. First, we can collect all the exact data values which can be further used to analyze, not just some aggregation results. Second, an anonymous function is realized to preserve data privacy in a high security level. Last, our approach is efficient and suitable for use.

Given the existing methods for preserving privacy of nodes in a crowdsensing system, our approach contributes to expanding the tool set for privacy preserving in that with our approach the server is able to access the original data, not just aggregated data.

A preliminary version of the work is reported in [39].

7. Conclusions

In this paper, we focus on the potential privacy risk of the mobile node in a mobile crowd sensing system. A mobile node's privacy may be compromised if its sensing data is associated with the node by a malicious attacker. Existing approaches try to protect node privacy by manipulating the sensing data sent to other nodes and the data center. Such approaches have a main drawback that the collected sensing data have a lower utility. We have proposed an approach

called P^3 for preserving node privacy. This approach determines a data forwarding path by which a source node sends its sensing data along the forwarding path. By using asymmetric encryptions, the source node is protected against malicious attackers which may try to trace back along the forwarding path. Through the security analysis based on the attack model, the possibility of malicious nodes compromising the privacy of data originators is low. Simulation results demonstrate that our approach has a low overhead of encryption computations and storage.

Notations

- u_i : The i th node
- d_{ij} : The j th data of node u_i
- D_i : The data set of node u_i
- P_{K_i} : The public key of node u_i
- S_{K_i} : The private key of node u_i
- S : The central server
- C : Data collection set in the server
- H : The public key handler
- P_i : The i th package be transferred by nodes
- t : The current time
- T : The forwarding time threshold
- n : The number of hops.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

The research is supported in part by Wenzhou Vocational & Technical College Research Project (no. WZY2017002), Wenzhou Science and Technology Bureau Program (no. G2017037), and Education Department of Zhejiang Province (2016 Educational Technology Research Project no. JB084).

References

- [1] T. Liu and Y. Zhu, "Social welfare maximization in participatory smartphone sensing," *Computer Networks*, vol. 73, pp. 195–209, 2014.
- [2] Y. Zhu, Y. Wang, G. Forman, and H. Wei, "Mining large-scale GPS streams for connectivity refinement of road maps," *The Computer Journal*, vol. 58, no. 9, pp. 2109–2119, 2014.
- [3] R. Jiang, Y. Zhu, X. Wang, and L. M. Ni, "TMC: Exploiting Trajectories for Multicast in Sparse Vehicular Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 1, pp. 262–271, 2015.
- [4] K. Wu, H. Li, L. Wang et al., "HJam: attachment transmission in WLANs," *IEEE Transactions on Mobile Computing*, vol. 12, no. 12, pp. 2334–2345, 2013.
- [5] K. Wu, H. Tan, Y. Liu, J. Zhang, Q. Zhang, and L. M. Ni, "Side channel: Bits over interference," *IEEE Transactions on Mobile Computing*, vol. 11, no. 8, pp. 1317–1330, 2012.

- [6] K. Wu, H. Tan, H.-L. Ngan, Y. Liu, and L. M. Ni, "Chip error pattern analysis in IEEE 802.15.4," *IEEE Transactions on Mobile Computing*, vol. 11, no. 4, pp. 543–552, 2012.
- [7] Y. Zhu, Z. Li, H. Zhu, M. Li, and Q. Zhang, "A compressive sensing approach to urban traffic estimation with probe vehicles," *IEEE Transactions on Mobile Computing*, vol. 12, no. 11, pp. 2289–2302, 2013.
- [8] Y. Zhu, X. Liu, M. Li, and Q. Zhang, "POVA: traffic light sensing with probe vehicles," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1390–1400, 2013.
- [9] Y. Zhu, Y. Bao, and B. Li, "On maximizing delay-constrained coverage of urban vehicular networks," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 4, pp. 804–817, 2012.
- [10] Y. Zhu, R. Jiang, J. Zhao, and L. M. Ni, "Correlating mobility with social encounters: distributed localization in sparse mobile networks," *Wireless Networks*, vol. 21, no. 1, pp. 201–215, 2015.
- [11] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni, "CSI-based indoor localization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1300–1309, 2013.
- [12] J. A. Burke, D. Estrin, M. Hansen et al., "Participatory sensing," *Center for Embedded Network Sensing*, 2006.
- [13] Z. Yang, S. Zhong, and R. N. Wright, "Privacy-preserving classification of customer data without loss of accuracy," in *Proceedings of the SDM*, pp. 92–102, 2005.
- [14] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregation in mobile sensing," in *Proceedings of the 2012 20th IEEE International Conference on Network Protocols, ICNP 2012*, USA, November 2012.
- [15] T. Abdelzaher, Y. Anokwa, P. Boda et al., "Mobiscopes for human spaces," *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 20–29, 2007.
- [16] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, and R. A. Peterson, "People-centric urban sensing," in *Proceedings of the 2nd Annual International Workshop on Wireless Internet (WICON '06)*, pp. 1–14, ACM, August 2006.
- [17] J. Corburn, "Confronting the Challenges in Reconnecting Urban Planning and Public Health," *American Journal of Public Health*, vol. 94, no. 4, pp. 541–546, 2004.
- [18] Z. Zhu and G. Cao, "APLAUS: a privacy-preserving location proof updating system for location-based services," in *Proceedings of the 30th IEEE International Conference on Computer Communications (IEEE INFOCOM 2011)*, pp. 1889–1897, Shanghai, China, April 2011.
- [19] Q. Li and G. Cao, "Mitigating routing misbehavior in disruption tolerant networks," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 664–675, 2012.
- [20] Q. H. Li, S. C. Zhu, and G. H. Cao, "Routing in socially selfish delay tolerant networks," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM '10)*, pp. 1–9, March 2010.
- [21] Q. Li and G. Cao, "Providing privacy-aware incentives for mobile sensing," in *Proceedings of the 11th IEEE International Conference on Pervasive Computing and Communications, PerCom 2013*, pp. 76–84, USA, March 2013.
- [22] E. De Cristofaro and C. Soriente, "Short paper: PEPSI: Privacy-Enhanced Participatory Sensing Infrastructure," in *Proceedings of the 4th ACM Conference on Wireless Network Security, WiSec'11*, pp. 23–28, Germany, June 2011.
- [23] Q. Li, W. Gao, S. Zhu, and G. Cao, "A routing protocol for socially selfish delay tolerant networks," *Ad Hoc Networks*, vol. 10, no. 8, pp. 1619–1632, 2012.
- [24] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC '09*, pp. 169–178, USA, June 2009.
- [25] C. Castelluccia, A. C.-F. Chan, E. Mykletun, and G. Tsudik, "Efficient and provably secure aggregation of encrypted data in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 5, no. 3, pp. 1–36, 2009.
- [26] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of Cryptography, vol. 3378 of Lecture Notes in Computer Science*, pp. 325–341, Springer, Berlin, Germany, 2005.
- [27] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: a secure hop-by-hop data aggregation protocol for sensor networks," *ACM Transactions on Information and System Security*, vol. 11, no. 4, article 18, 2008.
- [28] E. Shi, T.-H. H. Chan, E. G. Rieffel, R. Chow, and D. Song, "Privacy-preserving aggregation of time-series data," in *Proceedings of the NDSS*, p. 4, 2011.
- [29] T.-H. H. Chan, E. Shi, and D. Song, "Privacy-preserving stream aggregation with fault tolerance," in *Financial Cryptography and Data Security*, pp. 200–214, Springer, Berlin, Germany, 2012.
- [30] I. Boutsis and V. Kalogeraki, "Privacy preservation for participatory sensing data," in *Proceedings of the 11th IEEE International Conference on Pervasive Computing and Communications (PerCom '13)*, pp. 103–113, IEEE, March 2013.
- [31] J. Shi, R. Zhang, Y. Liu, and Y. Zhang, "PriSense: privacy-preserving data aggregation in people-centric urban sensing systems," in *Proceedings of the IEEE INFOCOM*, San Diego, Calif, USA, March 2010.
- [32] D. Agrawal and C. C. Aggarwal, "On the design and quantification of privacy preserving data mining algorithms," in *Proceedings of the ACM PODS*, pp. 247–255, 2001.
- [33] A. Evfimievski, J. Gehrke, and R. Srikant, "Limiting Privacy Breaches in Privacy Preserving Data Mining," in *Proceedings of the ACM PODS*, pp. 211–222, June 2003.
- [34] R. K. Ganti, N. Pham, Y.-E. Tsai, and T. F. Abdelzaher, "Pool-View: Stream privacy for grassroots participatory sensing," in *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems, SenSys 2008*, pp. 281–294, USA, November 2008.
- [35] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *Proceedings of the IEEE ICDM*, pp. 99–106.
- [36] F. Zhang, L. He, W. He, and X. Liu, "Data perturbation with state-dependent noise for participatory sensing," in *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 2012*, pp. 2246–2254, USA, March 2012.
- [37] J. Horey, M. M. Groat, S. Forrest, and F. Esponda, "Anonymous data collection in sensor networks," in *Proceedings of the 4th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous '07)*, pp. 1–8, August 2007.
- [38] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Communications of the ACM*, vol. 42, no. 2, pp. 39–41, 1999.
- [39] Q. Chen, S. Zheng, and Z. Weng, "Data collection with privacy preserving in participatory sensing," in *Proceedings of the IEEE ICPADS*, 2017.