Efficient and Privacy-Preserving Dynamic Spatial Query Scheme for Ride-Hailing Services

Fengwei Wang, *Student Member, IEEE*, Hui Zhu, *Member, IEEE*, Ximeng Liu, *Member, IEEE*, Rongxing Lu, *Senior Member, IEEE*, Fenghua Li, Hui Li, *Member, IEEE*, and Songnian Zhang

***Abstract*—With the prosperity of mobile internet and the pervasiveness of location-aware mobile terminals, online ride-hailing, a high-level location-based service (LBS) which relies on dynamic spatial query, has made our life more convenient. However, the flourish of ride-hailing service still faces many severe challenges since users' location privacy and service provider's data security. In this paper, we present an efficient and privacy-preserving dynamic spatial query scheme (TRACE) for ride-hailing service. With TRACE, users (i.e., consumers and vehicles) can access ride-hailing service without divulging their sensitive location information, meanwhile, the ride-hailing server can achieve the necessary commercial operating information while keeping its sensitive data (i.e., the space division information) confidential. Specifically, with two proposed efficient and secure spatial query algorithms, named FSSQ and ESVQ, all location-related data are encrypted by its owner before being sent out, and are calculated without decryption during the spatial query process. Therefore, consumers, vehicles, and service provider cannot obtain each other's sensitive information. Detailed security analysis shows that TRACE can resist various known security threats. Furthermore, TRACE is implemented in the real environment, and extensive simulation results over smart phones demonstrate that the scheme is highly efficient and can be implemented effectively.**

***Index Terms*—Location-based service (LBS), online ride-hailing, privacy-preserving, spatial query.**

Manuscript received March 3, 2018; revised July 12, 2018; accepted August 20, 2018. Date of publication September 5, 2018; date of current version November 12, 2018. The work of H. Zhu was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802201, in part by the National Natural Science Foundation of China under Grants 61672411, U1401251, and 81600574, in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2016ZDJC-04, and in part by China 111 Project (B16037). The work of R. Lu was supported in part by the Natural Sciences and Engineering Research Discovery (Rgpin 04009), in part by the NBIF StartUp (Rif 2017- 915012), in part by the URF (Nf-2017-05), and in part by the HMF (2017 Ys-4). The work of X. Liu was supported by the National Natural Science Foundation of China under Grant 61702105. The review of this paper was coordinated by Prof. L. Guo. *(Corresponding author: Hui Zhu.)*

F. Wang, H. Zhu, H. Li, and S. Zhang are with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: xdwangfengwei@gmail.com; zhuhui@xidian.edu.cn; lihui@mail.xidian.edu.cn; ruisijundao_zsn@163.com).

X. Liu is with the School of Information Systems, Singapore Management University, Singapore 188065 (e-mail: snbnix@gmail.com).

R. Lu is with Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).

F. Li is with the State Key Laboratory of Integrated Services Networks, Institute of Information Engineering and Chinese Academic of Sciences, Beijing 100864, China (e-mail: lfh@iie.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2018.2868869

I. Introduction

NOWADAYS, the online ride-hailing service, one of a burgeoning location based services (LBSs), has gained great popularity all over the world [1]–[3]. Compared with the traditional vehicles such as subways and buses, the ride-hailing service can make the trip of passengers more convenient and flexible [4]. As shown in Fig. 1, the ride-hailing service system commonly consists of three parts: (1) ride-hailing server; (2) registered customers; and (3) registered vehicles. To hail a vehicle, a consumer only needs to choose her/his desired pick-up location on the map, and sends the hailing request to the ride-hailing server, then the request will be forwarded to the vehicles around the pick-up point. For providing more user-friendly service, the ride-hailing server generally divides an area into subregions for analyzing the distributions of ride-hailing behaviors, and detecting the density of vehicles on the map dynamically, which allows the service provider to offer scheduling information for vehicles in real time.

However, owing to the sensitivity of users' locations and the data stored in service provider, there are still many severe challenges lying ahead the ride-hailing service [5]–[11]. On the one hand, once location information of ride-hailing users (including costumers and vehicles) is obtained by attackers, it leads to many computer-aided crime possibilities (harassment, cat theft, kidnapping, etc.). On the other hand, for reducing ride-hailing time of users and offering a better service, the service server will analyze the distribution of ride-hailing behaviors on the map, and detect the density of vehicles via dividing the hailing space into subregions, moreover, the service provider optimizes the space division continuously with the data collected from users. Disclosure of the space division information may reveal trade secrets directly, which brings an economic loss of service provider. Therefore, during the ride-hailing service, the accurate locations of consumers and vehicles cannot be leaked, and the space division information of service provider should be kept confidential. Nevertheless, most traditional ride-hailing services rely on the fact that users provide accurate locations for the service provider, which exists the risk of data leakage. Thus, how to design a secure and efficient privacy-preserving dynamic spatial query scheme for ride-hailing service system has attracted considerable interest recently.

To address these challenges, many location privacy-preserving schemes for LBS have been proposed, which mainly relies on anonymization [12], [13] and cryptographic

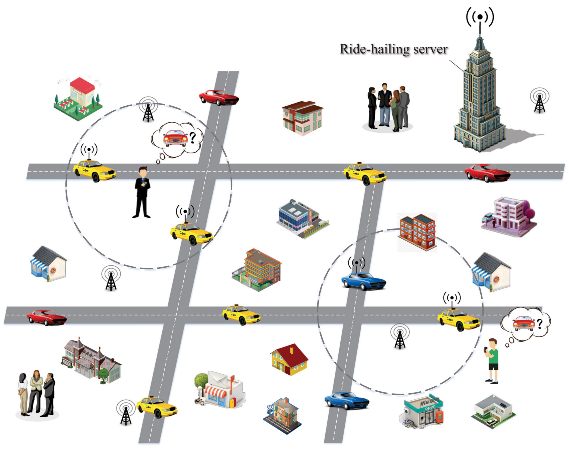


Fig. 1. Conceptual architecture of car-hailing service.

techniques [14], [15]. Specifically, anonymization-based location privacy-preserving schemes commonly blur a user's accurate location into an anonymous region which contains other $k-1$ users, and the locations of the k users are indistinguishable. Therefore, the probability of identifying the user's location is not greater than $1/k$. However, if the k users are in a same sensitive location, such as a secret meeting room, their location privacy may also be leaked. Meanwhile, anonymization technique brings heavy communication overhead to the user's side, which causes great energy consumption on the mobile devices. Furthermore, cryptographic-based location privacy-preserving schemes are generally constructed with homomorphic encryption or secure multi-party computation (SMC), which encrypt the users' accurate locations in clients, and allow to generate accurate spatial query results over the encrypted location data, but in general, most of them contain massive complicated arithmetical operations, which makes them unsuitable to be used in mobile terminals. Therefore, most existing location privacy-preserving schemes can protect users' location privacy in some degree, but they are difficult to be applied in online ride-hailing service system.

In this paper, we propose an efficien$\underline{\text{t}}$ and p$\underline{\text{r}}$ivacy-preserving dyn$\underline{\text{a}}$mi$\underline{\text{c}}$ spatial qu$\underline{\text{e}}$ry scheme for ride-hailing service, named TRACE. With TRACE, the ride-hailing server can find the subregions where the vehicles, or the pick-up/take-off points of customers reside, while the sensitive location-related information of ride-hailing server and users is well protected. Moreover, a customer can query the vehicles around her/his pick-up point efficiently, while the sensitive query range of the customer and accurate locations of vehicles are kept confidential. Specifically, the main contributions of this paper are threefold.

- *First*, TRACE provides a privacy-preserving dynamic spatial query scheme for ride-hailing service. With TRACE, the accurate locations of consumers and vehicles, and the space division information of service providers can be well protected. Concretely, before being sent out, all of the sensitive location-related data are masked into chipertexts by its owner, and are calculated without decryption during the spatial query process, therefore, consumers, vehicles, and ride-hailing server cannot obtain each other's sensitive information. In addition, even if attackers can eavesdrop on all data packets transmitted among users and the server, they cannot achieve available information.
- *Second*, TRACE achieves accurate spatial query. Based on quadtree data structure and lightweight multiparty random masking technique, we construct a fast and secure subregion query (FSSQ) algorithm, which allows the service provider to find the subregions where the vehicles, or the pick-up/take-off points of customers reside preciously while protecting users' locations and ride-hailing server's space division privacy. Meanwhile, based on FSSQ, another efficient and secure vehicle query (ESVQ) algorithm is proposed for consumers to accurately search the vehicles around the pick-up point without leaking the accurate location data of consumers and vehicles to each other.
- *Third*, TRACE is high-efficiency in terms of computation complexity and communication overhead. Our proposed TRACE is a lightweight privacy-preserving spatial query scheme which mainly relies on addition and multiplication operations, therefore, the computation efficiency of TRACE can be ensured. Meanwhile, based on quadtree data structure, the time of spatial query is greatly reduced. Moreover, the proposed TRACE is equipped with BLS short signature technique [16], which improve the communication efficiency. In order to evaluate the effectiveness of TRACE, we develop a demo application, and test through smart phones and PC with a real dataset. Extensive results show that TRACE is effective in the real environment.

The remainder of this paper is organized as follows. In Section II, we formalize the system model, security requirements, and identify our design goal. In Section III, we review the bilinear pairing, quadtree data structure, efficient and privacy-preserving cosine similarity computing protocol, and the strategy of convex point in polygon as the preliminaries. Then, we propose our TRACE in Section IV, followed by the security analysis and performance evaluation in Section V and Section VI, respectively. We also review some related works in Section VII. Finally, we draw our conclusions in Section VIII.

II. Models, Security Requirements and Design Goal

In this section, we formalize the system model and security requirements, and identify our design goal.

A. System Model

In our system model, we mainly focus on how to provide an accurate and efficient privacy-preserving ride-hailing service for users and service provider. Each user, such as a consumer or a vehicle, is equipped with a smartphone, which can connect with the service server for achieving ride-hailing service. Specifically, as mentioned-above, our system consists of three parts: 1) ride-hailing server (RS); 2) registered consumer (RC) and 3) registered vehicle (RV). As shown in Fig. 2.

- RS is the server of ride-hailing service provider, which mainly performs three functions: a) detect the distribution of ride-hailing behaviors and the density of registered vehicles in real time, b) forward vehicles query and hailing

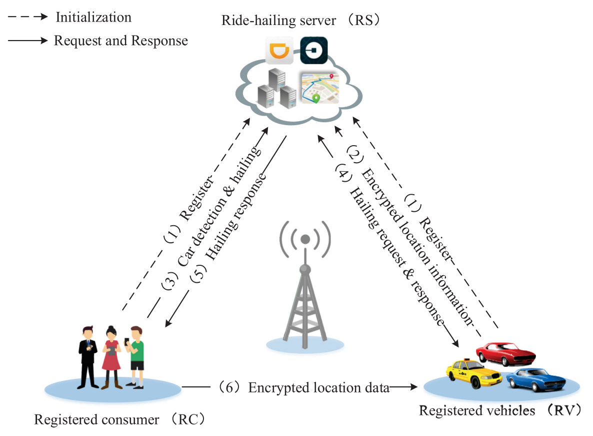


Fig. 2. System model under considered.

requests/responses among users, and c) provide scheduling information for vehicles. In our system, RS dynamically queries the subregions where the vehicles and consumers' desired pick-up/take-off points reside with encrypted location data to achieve these three functions.

- RC is a customer who has registered in RS. Based on ride-hailing applications, RC can choose her/his desired pick-up/take-off locations on the map, and send hailing request to RS, then the hailing request will be forwarded to the vehicles around the pick-up point by RS.
- RV is a vehicle which has registered in RS. In the process of ride-hailing service, RV is responsible for submitting its masked location information to RS at regular intervals, which is used for the dynamic subregion query for RS. In addition, RV can access the scheduling information from RS.

B. Security Requirement

Protecting the location privacy of RC and RV, and the space division information of RS is crucial during the service provider. In our security model, we consider that RS is *credible-but-greedy*, and RC/RV is *honest-but-curious*. Specially, RS honestly executes the operations over chipertext, and forwards the hailing requests and responses reliably, but it is curious about the accurate location information of RC and RV. RC and RV do not send false information, but RC tries to analyze the accurate locations of RVs through hailing responses, and RV wants to obtain RCs' locations or pick-up points via the hailing requests. Moreover, both RC and RV want to obtain RS's space division information during the spatial query process. Furthermore, attackers may tamper and falsify the data, or disguise legitimate users for accessing service. Considering above security issues, the following security requirements should be satisfied.

- *Privacy:* Protecting a user's location information from RS and other users, i.e., during the ride-hailing service, RCs' pick-up/take-off points should be kept secret from RS and un-hired RVs, and RVs' accurate locations cannot be leaked to RS and RCs. That is, even if RS can obtain all the hailing requests and responses form RCs and RV, it can only detect the approximate regions where RV and RCs' pick-up/take-off points reside, rather than the accurate locations.
- *Confidentiality:* Ensuring the security of RS's sensitive data during the ride-hailing service. To obtain the density of RVs and the distribution of ride-hailing behaviors, RS needs to divide an area into subregions, and optimize the space division continuously via collecting and analyzing the location data of users. The leakage of the space division information may reveal trade secrets, which brings a direct economics of ride-hailing service provider. Therefore, during the spatial query process, the space division information cannot be leaked.
- *Authentication:* Authenticating that encrypted hailing requests and responses are really sent by legal users and not modified during the transmission, i.e., if an illegal user disguises a legal RV for stealing a RC's pick-up point data, the malicious operation should be prevented. Moreover, all of the packets transmitted among RC, RV, and RS should be authenticated so that users can access authentic and reliable ride-hailing service.

C. Design Goal

Under the aforementioned system model and security requirements, our design goal is to develop an efficient and privacy-preserving dynamic spatial query scheme for ride-hailing service. Specifically, the following three objectives should be achieved.

- *Guarantee security and privacy-preserving:* If TRACE does not consider the security issues, user's and service provider's sensitive information could be disclosed, which leads to serious consequences. Then, the ride-hailing service cannot step into its flourish. Thus, TRACE should achieve the confidentiality and authentication simultaneously.
- *Achieve accuracy of dynamic spatial query results:* Users' experience is a crucial aspect of the proposed scheme. In our system, dynamic spatial query is the fundamental function of ride-hailing service, which impacts user experience directly. Therefore, it is significant that the precision of spatial query cannot be lowered while protecting privacy of the server and users.
- *Low computation complexity and communication overhead:* Although the performance of smartphones is continuously improved today, their batteries are still limited. Moreover, considering the real-time requirements of ride-hailing service and huge amount of users, the proposed scheme should consider the effectiveness in terms of computation and communication to reduce the power consumption for both ride-hailing servers and smartphones.

III. Preliminaries

In this section, we review the bilinear pairing [16], quadtree data structure [17], efficient and privacy-preserving cosine similarity computing protocol [18], and cross products (point in convex polygon strategies) [19]. These will serve as the basis of our TRACE.

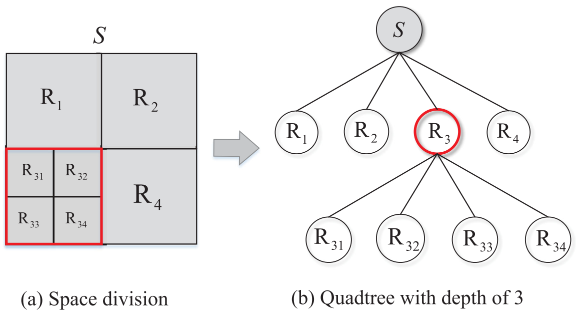


Fig. 3. Definition of quadtree.

A. Bilinear Pairing

Let $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ be two cyclic groups with the same prime order q, and g is a generator of group $\mathbb{G}$. Suppose $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ are equipped with a pairing, i.e., a non-degenerated and efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_{\mathbb{T}}$ has the following properties.

1) Bilineariry: $\forall g, h \in \mathbb{G}$, and $\forall a, b \in \mathbb{Z}_{\mathbb{q}}$, we have $e(g^a, h^b) = e(g, h)^{ab}$.

2) Nondegeneracy: $\exists$ at least one g, h, where $g, h \in \mathbb{G}$, which satisfies the condition that $e(g, h) \neq 1_{\mathbb{G}_{\mathbb{T}}}$.

3) Computable: $\forall g, h \in \mathbb{G}$, there is an efficient algorithm to compute $e(g, h)$.

B. Quadtree Data Structure

A quadtree is a tree data structure, which is widely used in spatial query. Concretely, a quadtree represents a partition of a two-dimensional space by decomposing the area into quadrants, subquadrants, and so on. In a quadtree, each non-leaf node has exactly four children, and each leaf node contains the data corresponding a specific subregion. The height of quadtree is dependent on the space division fineness requirements in spatial query, i.e., a quadtree with the depth of n is used to represent dividing an area into 4^{n-1} subregions, an example of a quadtree with the depth of 3 is shown in Fig. 3.

Given a two-dimensional space S and a point p in S, the procedure for querying the subregion of p is as follows.

- *Step1:* Divide S into subregions with quadtree data structure.
- *Step2:* Staring at the root node, check the child node where the point p is located, then recurse into this child node.
- *Step3:* When reaching the leaf node, return this leaf node, which represent the approximate region of point p.

C. Efficient and Privacy-Preserving Cosine Similarity Computing Protocol

Given a vector of P_A, $\vec{a} = (a_1, a_2, \ldots, a_n) \in \mathbb{F}_q^n$ and a vector of P_B, $\vec{b} = (b_1, b_2, \ldots, b_n) \in \mathbb{F}_q^n$, we can directly calculate the cosine similarity $\cos(\vec{a}, \vec{b})$ in an efficient and privacy-preserving way. The main calculation process is as follows.

- *Step1:* (performed by P_A) Given security parameters k_1, k_2, k_3, k_4, choose two large primes α, p such that $|p| = k_1$, $|\alpha| = k_2$, set $a_{n+1} = a_{n+2} = 0$. Choose a large random $s \in \mathbb{Z}_p$ and $n+2$ random numbers $|c_i| = k_3$, $i = 1, 2, \ldots, n+2$. Then P_A calculates
$$C_i = \begin{cases} s(a_i \cdot \alpha + c_i) \bmod p, & a_i \neq 0; \\ s \cdot c_i \bmod p, & a_i = 0; \end{cases}$$
and $A = \sum_{i=1}^{n} a_i^2$. What's more, P_A should keep $s^{-1} \bmod p$ secret. After these operations, $< \alpha, p, C_1, \ldots C_{n+2} >$ is sent to P_B.
- *Step2:* (performed by P_B) Set $b_{n+1} = b_{n+2} = 0$, select random numbers $|r_i| = k_4$, then calculate
$$D_{i=} \begin{cases} b_i \cdot \alpha \cdot C_i \bmod p, & b_i \neq 0; \\ r_i \cdot C_i \bmod p, & b_i = 0; \end{cases}$$
$B = \sum_{i=1}^{n} b_i^2$ and $D = \sum_{i=1}^{n+2} D_i \bmod p$. Then send $< B, D >$ back to P_A.
- *Step3:* (performed by P_A) Compute $E = s^{-1} \cdot D \bmod p$, $\vec{a} \cdot \vec{b} = \sum_{i=1}^{n} a_i \cdot b_i = ([E - (E \bmod \alpha^2)]/\alpha^2)$ and $\cos(\vec{a}, \vec{b}) = (\vec{a} \cdot \vec{b}/(\sqrt{A} \cdot \sqrt{B}))$.

During the above calculation, it can be figured that the vectors of P_A and P_B are confidential to each other.

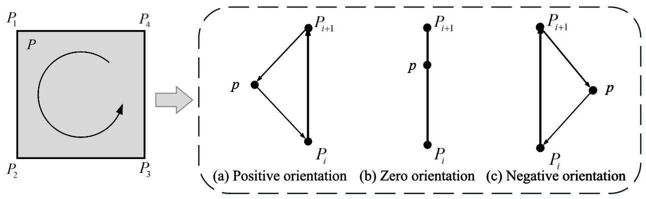


Fig. 4. Orientation of point p and polygon vertexes.

D. Cross Products - Point in Convex Polygon Strategy

Given a convex polygon P with n edges and a point p, the vertices $P_1 P_2 \ldots P_n$ are defined in anticlockwise direction. Assume that the coordinates of the vertexes and the point are defined as $< (x_1, y_1), (x_2, y_2), \ldots, (x_i, y_i), (x_{i+1}, y_{i+1}), \ldots, (x_n, y_n) >$ and (x_s, y_s), respectively. The point in convex polygon strategy is the protocol to determine whether the point p is within the convex polygon P. We can solve this problem by calculating points orientation [19]. As shown in Fig. 4, the triple points $< P_{i+1}, p, P_i >$ consist of two vertices of the polygon and a point p, we defined their orientations as follows.

- Positive orientation: $< P_{i+1}, p, P_i >$ is a counterclockwise turn.
- Negative orientation: $< P_{i+1}, p, P_i >$ is a clockwise turn.
- Zero orientation: $< P_{i+1}, p, P_i >$ is collinear.

The orientation of the $< P_{i+1}, p, P_i >$ can be computed as follows.

$$S_i = \begin{vmatrix} x_{i+1} & y_{i+1} & 1 \\ x_s & y_s & 1 \\ x_i & y_i & 1 \end{vmatrix}$$
$$= (x_s \cdot y_i + y_s \cdot x_{i+1} + x_i \cdot y_{i+1}) - (x_s \cdot y_{i+1} + y_s \cdot x_i + x_{i+1} \cdot y_i)$$

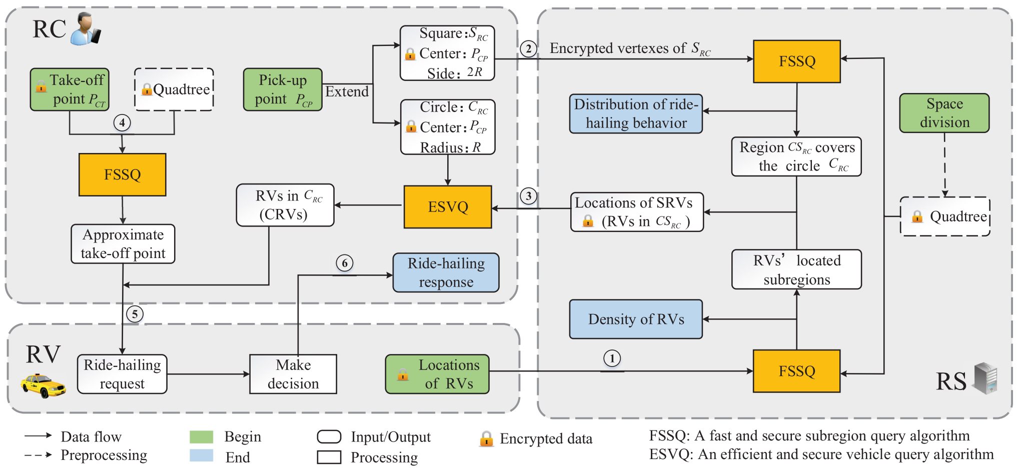

RC
Take-off point P_{CT}
Quadtree
Pick-up point P_{CP}
Extend
Square: S_{RC} Center: P_{CP} Side: $2R$
Circle: C_{RC} Center: P_{CP} Radius: R
FSSQ
ESVQ
RVs in C_{RC} (CRVs)
Approximate take-off point
Ride-hailing response
RV
Ride-hailing request
Make decision
Locations of RVs
Encrypted vertexes of S_{RC}
Distribution of ride-hailing behavior
Region CS_{RC} covers the circle C_{RC}
Locations of SRVs (RVs in CS_{RC})
RVs' located subregions
Density of RVs
Space division
Quadtree
RS
① ② ③ ④ ⑤ ⑥
Data flow
Preprocessing
Begin
End
Input/Output
Processing
Encrypted data
FSSQ: A fast and secure subregion query algorithm
ESVQ: An efficient and secure vehicle query algorithm

Fig. 5. Overview of TRACE.

Next, for the given convex polygon P and point p, whether the point is within the convex polygon can be determined by the following protocol.

- Let $i \in \{1, 2, \ldots, n\}$, $i' = (i + 1) \bmod n$, then compute S_i of the triple points $< P_{i'}, p, P_i >$, in which the vertex P_i is visited in an anticlockwise order.
- If all $S_i \geq 0$, the point p is within the convex polygon P; else, point p is outside the convex polygon P.

IV. Proposed Privacy-Preserving Scheme

In this section, we present our TRACE scheme, which mainly consists of three phases: 1) *system initialization;* 2) *dynamic and privacy-preserving subregion query;* and 3) *secure ride-hailing service*. The overview of TRACE is described in Fig. 5. After the system initialization, RS divides a two-dimensional hailing space into subregions which satisfies the quadtree data structure, and executes the FSSQ algorithm with RV and RC for querying the subregions where RV and RC's pick-up/take-off points reside. When accessing ride-hailing service, RC first chooses pick-up/take-off points on the map, and queries RVs around the pick-up point, then RC sends the ride-hailing request to these RVs through RS. During this process, RS can generate scheduling information for RVs through analyzing the distribution of ride-hailing behaviors in the whole ride-hailing space. To describe TRACE more clear, we give the description of used notations in Table I.

A. System Initialization

RS first chooses security parameter κ to generate the bilinear parameters $(q, g, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e)$ through running $Gen(\kappa)$. Then, RS chooses a random number in $\mathbb{Z}_q^*$ as its private key SK_{RS}, and computes its public key $PK_{RS} = g^{SK_{RS}}$. In addition, RS selects security parameters k_1, k_2, k_3, k_4, where $k_4 + 2k_2 < k_1, k_2 + k_3 < k_1, k_3 + k_4 < k_2$, a symmetric encryption $E()$, i.e., AES, and a secure cryptographic hash function $H()$, where $H : \{0, 1\}^* \to \mathbb{G}$, After this, RS chooses two large primes such that $|p| = k_1$, $|\alpha| = k_2$, a random number $s \in \mathbb{Z}_p^*$, and random numbers $|a_{ji}| = k_3$, where $i = 1, 2, \ldots, 6; j = 1, 2, \ldots, 4$. Finally, RS keeps its private key SK_{RS} secret, and publishes the system parameters $< q, g, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e, k_1, k_2, k_3, k_4, PK_{RS}, E(), H() >$.

When registering in RS, a ride-hailing consumer RC chooses a random number in $\mathbb{Z}_q^*$ as its private key SK_{RC_k}, and computes its corresponding public key $PK_{RC_k} = g^{SK_{RC_k}}$, then, RC chooses two large primes such that $|p'| = k_1, |\alpha'| = k_2$, a random number $s' \in \mathbb{Z}_p^*$, and random numbers $|d_i| = k_3$, where $i = 1, 2, \ldots, 4$. Finally, RC submits her/his public key

TABLE I
Definition of Notations in TRACE

Notation	Definition
$\kappa, k_1, k_2, k_3, k_4$	The security parameters of TRACE.
$q, g, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e$	The parameters of bilinear paring.
α, p	Two large primes set by RS.
α', p'	Two large primes set by RC.
s, s', a_{ji}, d_i	Random numbers used in masking region data, i.e., the divided subregions of RS.
r_k	Random numbers used in masking RV's location.
sk	The session keys among RS, RC and RV, i.e., sk_{SC_k} is the session key between RC and RS.
N	The quadtree of space division.
EN	The encrypted quadtree of space division.
(x_{Nij}, y_{Nij})	The j^{th} vertex of quadtree node N_i.
(x_{CP}, y_{CP})	RC's desired pick-up point.
(x_{CT}, y_{CT})	RC's desired take-off point.
(x_V, y_V)	RV's location coordinate.
$A, C_1, \cdots, C_5$	Aggregated data from locations and the encrypted quadtree.
C_{RC}	The circle range around the pick-up point (x_{CP}, y_{CP}).
S_{RC}	The square range covers C_{RC}.
CS_{RC}	The region consists of 4 quadtree nodes covering S_{RC}.
SRV	The RV in CS_{RC}.
(x_{SV}, y_{SV})	The location of SRV.
VQP	The vehicle query information of RC.
CRV	The RV in C_{RC}.
ATP	The approximately take-off point location randomly selected by RS.
$E(\cdot)$	The secure symmetric encryption algorithm, i.e., AES.
$H(\cdot)$	The secure cryptographic hash function.
$\pi(\cdot)$	A random permutation function.

PK_{RC_k} to RS, and negotiates the session key $sk_{SC_k} = PK_{RS}{}^{SK_{RC_k}} = g^{SK_{RS} \cdot SK_{RC_k}}$ with RS.

Similarly, when registering in RS, a ride-hailing vehicle RV first generates its private key SK_{RV_k} and corresponding public key $PK_{RV_k} = g^{SK_{RV_k}}$, moreover, RV selects random numbers $|r_k| = k_4$, where $k = ij$, i is the number of nodes of quadtree; $j = 1, 2, \ldots, 4$ in FSSQ, and $k = 1, 2, \ldots, 5$ in ESVQ. Finally, RV sends the public key PK_{RV} to RS, and negotiates the session key $sk_{SV_k} = PK_{RS}{}^{SK_{RV_k}} = g^{SK_{RS} \cdot SK_{RV_k}}$ with RS.

B. Dynamic and Privacy-Preserving Subregion Query

In this phase, the space division data of RS and the accurate locations of users are masked via adding random numbers s, a_{ji} and r_{ij}, thus, the sensitive information of RS and users can be well protected. Meanwhile, through eliminating the superfluous random numbers in the query result, RS can find the subregions where the RVs, or the pick-up/take-off points of RCs reside preciously.

1) Encrypted Space Division Data Generation: With quadtree data structure, RS first divides a two-dimensional hailing space into subregions with squares or rectangles. Assume that the number of quadtree nodes is m, the quadtree can be presented by an array $N = \{N_1, N_2, \ldots, N_m\}$. Obviously, each tuple N_i represents a square or rectangle subregion, and the vertexes stored in N_i can be defined as follows:

$$< (x_{Ni1}, y_{Ni1}), (x_{Ni2}, y_{Ni2}), (x_{Ni3}, y_{Ni2}), (x_{Ni4}, y_{Ni4}) >,$$

where $i = 1, 2, \ldots, m$.

RS chooses two adjacent vertexes (x_{Nij}, y_{Nij}) and $(x_{Nij'}, y_{Nij'})$ of N_i in anticlockwise direction, and executes the following calculations to mask the coordinates:

$$EN_{ij1} = s(x_{Nij} \cdot \alpha + a_{j1}) \bmod p$$
$$EN_{ij2} = s(y_{Nij} \cdot \alpha + a_{j2}) \bmod p$$
$$EN_{ij3} = s(x_{Nij'} \cdot \alpha + a_{j3}) \bmod p$$
$$EN_{ij4} = s(y_{Nij'} \cdot \alpha + a_{j4}) \bmod p$$
$$EN_{ij5} = s(x_{Nij} \cdot y_{Nij'} \cdot \alpha + a_{j5}) \bmod p$$
$$EN_{ij6} = s(x_{Nij'} \cdot y_{Nij} \cdot \alpha + a_{j6}) \bmod p,$$

where $i=1, 2, \ldots, m$, $j=1, 2, \ldots, 4$, and $j'=(j+1) \bmod 4$.

Then RS computes $EN_{ij} = EN_{ij1} \parallel EN_{ij2} \parallel EN_{ij3} \parallel EN_{ij4} \parallel EN_{ij5} \parallel EN_{ij6}$, and $EN_i = EN_{i1} \parallel EN_{i2} \parallel EN_{i3} \parallel EN_{i4}$ to obtain the encrypted quadtree $EN = \{EN_1, EN_2, \ldots, EN_m\}$. After this, RS creates the signature $Sig_{RS} = H(\alpha \parallel p \parallel EN \parallel RS \parallel TS \parallel SI)^{SK_{RS}}$, where TS is the time stamp, and SI is the session ID to resist the potential replay attack.

Finally, RS keeps $s^{-1} \bmod p$ secret, and sends $E_{sk_{users}}(\alpha \parallel p \parallel EN \parallel RS \parallel TS \parallel SI \parallel Sig_{RS})$ to RCs and RVs, where sk_{users} represent the session keys between RS and users (RCs and RVs). Specially, the encrypted quadtree EN will be stored in users' client until the space division is updated.

2) Encrypted Vehicle's Location Submission: After receiving $E_{sk_{SV_k}}(\alpha \parallel p \parallel EN \parallel RS \parallel TS \parallel SI \parallel Sig_{RS})$ from RS, RV first decrypts it with the session key sk_{SV_k} to obtain $\alpha \parallel p \parallel EN \parallel RS \parallel TS \parallel SI \parallel Sig_{RS}$, and checks TS, SI and Sig_{RS} to verify its validity, i.e., verifying whether $e(g, Sig_{RS}) = e(PK_{RS}, H(\alpha \parallel p \parallel EN \parallel RS \parallel TS \parallel SI))$. If it does hold, the packet is valid. Assume that the location of RV is (x_V, y_V), then the follow operations are executed by RV:

$$A_{ij1} = r_{ij} \cdot \alpha(x_V \cdot EN_{ij4} + y_V \cdot EN_{ij1} + EN_{ij6}) \bmod p$$
$$A_{ij2} = r_{ij} \cdot \alpha(x_V \cdot EN_{ij2} + y_V \cdot EN_{ij3} + EN_{ij5}) \bmod p,$$

where $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, 4$.

After that, for each $j = 1, 2, \ldots, 4$, RV obtains $A_{ij} = A_{ij1} \parallel A_{ij2}$, and runs random permutation function $\pi(A_{ij})$ to make the order of j chaotic. Then, RV computes $A_i = A_{i1} \parallel A_{i2} \parallel A_{i3} \parallel A_{i4}$, $A = \{A_1, A_2, \ldots, A_m\}$, and creates the signature $Sig_{RV_k} = H(A \parallel RV \parallel TS \parallel SI)^{SK_{RV_k}}$. Finally, RV sends $E_{sk_{SV_k}}(A \parallel RV \parallel TS \parallel SI \parallel Sig_{RV_k})$ to RS at regular intervals (i.e., every 30 seconds) to achieve dynamic subregion query for RS.

3) Dynamic Vehicles Subregion Query: After receiving $E_{sk_{SV_k}}(A \parallel RV \parallel TS \parallel SI \parallel Sig_{RV_k})$, RS first decrypts it with the session key sk_{SV_k} to obtain $A \parallel RV \parallel TS \parallel SI \parallel Sig_{RV_k}$, and checks TS, SI and Sig_{RV_k} to verify its validity. Note that $\{A_1, A_2, \ldots, A_m\}$ is corresponding to the m nodes of quadtree, then, RS performs the following operations to determine whether the RV is within the subregion which is presented by node N_i:

$$\begin{aligned} B_{ij1} &= s^{-1} \cdot A_{ij1} \bmod p \\ &= s^{-1} \cdot r_{ij} \cdot \alpha(x_V \cdot EN_{ij4} + y_V \cdot EN_{ij1} + EN_{ij6}) \bmod p \\ &= s^{-1} \cdot r_{ij} \cdot s[\alpha^2(x_V \cdot y_{Nij'} + y_V \cdot x_{Nij} + x_{Nij'} \cdot y_{Nij}) \\ &\quad + \alpha(x_V \cdot a_{j4} + y_V \cdot a_{j1} + a_{j6})] \bmod p \end{aligned}$$

$$\begin{aligned} B'_{ij1} &= \frac{B_{ij1} - B_{ij1} \bmod \alpha^2}{\alpha^2} \\ &= r_i(x_V \cdot y_{Nij'} + y_V \cdot x_{Nij} + x_{Nij'} \cdot y_{Nij}). \end{aligned}$$

$$\begin{aligned} B_{ij2} &= s^{-1} \cdot A_{ij2} \bmod p \\ &= s^{-1} \cdot r_{ij} \cdot \alpha(x_V \cdot EN_{ij2} + y_V \cdot EN_{ij3} + EN_{ij5}) \bmod p \\ &= s^{-1} \cdot r_{ij} \cdot s[\alpha^2(x_V \cdot y_{Nij} + y_V \cdot x_{Nij'} + x_{Nij} \cdot y_{Nij'}) \\ &\quad + \alpha(x_V \cdot a_{j2} + y_V \cdot a_{j3} + a_{j5})] \bmod p \end{aligned}$$

$$\begin{aligned} B'_{ij2} &= \frac{B_{ij2} - B_{ij2} \bmod \alpha^2}{\alpha^2} \\ &= r_i(x_V \cdot y_{Nij} + y_V \cdot x_{Nij'} + x_{Nij} \cdot y_{Nij'}). \end{aligned}$$

$$\begin{aligned} B_{ij} &= B'_{ij2} - B'_{i1} \\ &= r_i[(x_V \cdot y_{Nij} + y_V \cdot x_{Nij'} + x_{Nij} \cdot y_{Nij'}) \\ &\quad - (x_V \cdot y_{Nij'} + y_V \cdot x_{Nij} + x_{Nij'} \cdot y_{Nij})]. \end{aligned}$$

After that, RS obtains B_{ij}, where $j = 1, 2, \ldots, 4$. If all of the $B_{ij} \geq 0$, RS can determine that RV resides in N_i, otherwise,

Algorithm 1: FSSQ: Fast and Secure Subregion Query.

```
Input: The space division quadtree $\{N_1, N_2, \ldots, N_m\}$ of
    RS; Location coordinate of a point, take $(x_V, y_V)$ of
    RV as an example.
Output: The subregion where RV resides.
 1: RS encrypts the quadtree $\rightarrow \{EN_1, EN_2, \ldots, EN_m\}$;
 2: for $i = 1$ to $i = m$ do
 3:     RV computes $A_i$ with $EN_i$ and $(x_V, y_V)$;
 4: end for
 5: function judge$A_i$         ▷ Whether RV is within region $N_i$
 6:     for $j = 1$ to $j = 4$ do
 7:         RS computes $B_{ij}$;
 8:         if $B_{ij} < 0$ then
 9:             return $false$;     ▷ RV is outside region $N_i$
10:         end if
11:     end for
12:     return $true$;              ▷ RV is within region $N_i$
13: end function
14: for $root\ node$ to $leaf\ nodes$ do
15:     RS determines the nodes which RV resides with
16:     $judge(\cdot)$;
17:     return $N_i$;
18: end for
```

RV is outside this subregion. Finally, starting with the root node of quadtree $N = \{N_1, N_2, \ldots, N_m\}$, RS can query the subregions of RV until leaf nodes. Moreover, RS executes the above operations with all RVs to obtain the subregions of all RVs, and achieves the density of vehicles in the whole hailing space dynamically.

Correctness of the FSSQ: Based on the relationship of k_1, k_2, k_3, and k_4 presented above, it is obvious that FSSQ satisfied the constraints $r_i[\alpha^2(x_V \cdot y_{Nij'} + y_V \cdot x_{Nij} + x_{Nij'} \cdot y_{Nij}) + \alpha(x_V \cdot a_{i4} + y_V \cdot a_{i1} + a_{i6})]$, $r_i[\alpha^2(x_V \cdot y_{Nij} + y_V \cdot x_{Nij'} + x_{Nij} \cdot y_{Nij'}) + \alpha(x_V \cdot a_{i2} + y_V \cdot a_{i3} + a_{i5})] < p$ and $r_i \cdot \alpha(x_V \cdot a_{i4} + y_V \cdot a_{i1} + a_{i6})$, $r_i \cdot \alpha(x_V \cdot a_{i2} + y_V \cdot a_{i3} + a_{i5}) < \alpha^2$. Since the values of coordinates are not very big, we can choose applicable security parameters easily (such as $k_1 = 512$, $k_2 = 160$, $k_3 = 75$ and $k_4 = 75$). Note that the expression $B_{ij} = r_i[(x_V \cdot y_{Nij} + y_V \cdot x_{Nij'} + x_{Nij} \cdot y_{Nij'}) - (x_V \cdot y_{Nij'} + y_V \cdot x_{Nij} + x_{Nij'} \cdot y_{Nij})]$, which is formed by two divisors, one is random r_i, and another is the cross product of $< P_{j'}, p, P_j >$. Since r_i is a positive number, the sign of the cross product is clear. Then RS can find out whether RV is within the polygon through orientations of $< P_{j'}, p, P_j >$, where $i = 1, 2, \ldots, 4$.

Specially, in order to obtain the distribution of ride-hailing behaviors, and improve the efficiency of the vehicles search, RS will query the subregions covering RCs' pick-up/take-off points via performing FSSQ with RCs, which will be introduced in the following concretely.

C. Secure Vehicles Query and Ride-Hailing Services

In this phase, the query range of RC and the accurate locations of RVs are masked via adding random numbers s', d_i and r_k. Thus, the sensitive information of RC and RV can be well protected. Meanwhile, through eliminating the superfluous random numbers in the query result, RC can query the RVs around the pick-up point accurately.

1) Vehicles Query Request Generation: RC first chooses a pick-up point (x_{CP}, y_{CP}) on the map, and sets a search radius R, which is 1 km minimum. Based on (x_{CP}, y_{CP}) and R, a circle query range C_{RC}, and a square S_{RC} which covers C_{RC} with side $2R$ can be generated. Assume that the vertexes of S_{RC} is presented by $< (x_{S1}, y_{S1}), (x_{S2}, y_{S2}), (x_{S3}, y_{S3}), (x_{S4}, y_{S4}) >$. Then, similarly with RV computing A, RC executes FSSQ to aggregate the 4 vertexes of S_{RC} and the encrypted quadtree EN to obtain $C = C_1 \parallel C_2 \parallel C_3 \parallel C_4$. Moreover, RC computes

$$D_1 = s'(x_{CP} \cdot \alpha' + d_1) \bmod p'$$
$$D_2 = s'(y_{CP} \cdot \alpha' + d_2) \bmod p'$$
$$D_3 = s' \cdot d_3 \bmod p'$$
$$D_4 = s' \cdot d_4 \bmod p'.$$

After that, RC obtains $D = D_1 \parallel D_2 \parallel D_3 \parallel D_4$, and computes $E = x_{CP}^2 + y_{CP}^2 - R^2$. Then, RC makes the signature $Sig_{RC_k} = H(\alpha' \parallel p' \parallel D \parallel E \parallel RC \parallel TS \parallel SI)^{SK_{RC_k}}$, and generates the vehicles query packet $VQP = \alpha' \parallel p' \parallel D \parallel E \parallel RC \parallel TS \parallel SI \parallel Sig_{RC_k}$. Then, RC creates the signature $Sig'_{RC_k} = H(C \parallel VSQ \parallel RC \parallel TS \parallel SI)^{SK_{RC_k}}$, and computes $E_{sk_{SC_k}}(C \parallel VSQ \parallel RC \parallel TS \parallel SI \parallel Sig'_{RC_k})$. Finally, RS keeps $s'^{-1} \bmod p'$ secret, and sends $E_{sk_{SC_k}}(C \parallel VSQ \parallel RC \parallel TS \parallel SI \parallel Sig'_{RC_k})$ to RS.

2) Distribution of Ride-Hailing Behaviors Analyzing: After receiving $E_{sk_{SC_k}}(C \parallel VSQ \parallel RC \parallel TS \parallel SI \parallel Sig'_{RC_k})$ from RC, RS first decrypts it with the session key sk_{SC_k} to obtain $C \parallel VSQ \parallel RC \parallel TS \parallel SI \parallel Sig'_{RC_k}$, then checks TS, SI and Sig'_{RC_k} to verify its validity, i.e., verifying whether $e(g, Sig'_{RC_k}) = e(PK_{RC}, H(C \parallel VSQ \parallel RC \parallel TS \parallel SI)$. If it does hold, the packet is valid. After this, RS executes FSSQ with C to obtain 4 quadtree nodes where the 4 vertexes of S_{RC} reside, as a result, these 4 nodes can construct the region CS_{RC}, which covers the square S_{RC}. Specially, through collecting and analyzing CS_{RC} of each RC's query request on the map at regular intervals, RS can achieve dynamic distributions of ride-hailing behaviors, and generate the scheduling information for RVs. Moreover, it is obvious that RS knows the distribution of all RVs, therefore, RS can obtain the RVs in CS_{RC}, represented by SRVs. Finally, RS calculates $E_{sk_{SRV_s}}(VSQ)$, and forwards $E_{sk_{SRV_s}}(VSQ)$ to SRVs, where sk_{SRV_s} present the session keys between RS and SRVs.

3) Vehicles Query Response Generation: After receiving $E_{sk_{SRV_s}}(VSQ)$, each SRV first decrypts it with the session key sk_{SRV_k} to obtain $VQP = \alpha' \parallel p' \parallel D \parallel E \parallel RC \parallel TS \parallel SI \parallel Sig_{RC_k}$, then checks TS, SI and Sig_{RC_k} to verify its validity. In addition, the SRV calculates:

$$F_1 = x_{SV} \cdot \alpha \cdot D_1 \bmod p'$$
$$F_2 = y_{SV} \cdot \alpha \cdot D_2 \bmod p'$$

$$F_3 = r_3 \cdot D_3 \bmod p'$$
$$F_4 = r_4 \cdot D_4 \bmod p',$$

where $< x_{SV}, y_{SV} >$ is SRV's location.

After that, the SRV computes $F = r_5 \sum_{i=1}^{i=4} F_i$, $I = r_5(x_{SV}^2 + y_{SV}^2 + E)$, and creates the signature $Sig_{SRV_k} = H(I \parallel F \parallel SRV \parallel TS \parallel SI)^{SK_{SRV_k}}$. Finally, the SRV sends $E_{sk_{SRV_k}}(I \parallel F \parallel SRV \parallel TS \parallel SI \parallel Sig_{SRV_k})$ to RS. RS will decrypt it, and forwards $E_{sk_{SC_k}}(I \parallel F \parallel SRV \parallel TS \parallel SI \parallel Sig_{SRV_k})$ to RC.

4) Vehicles Query Result Reading: After receiving $E_{sk_{SC_k}}(I \parallel F \parallel SRV \parallel TS \parallel SI \parallel Sig_{SRV_k})$, RC first decrypts it with the session key sk_{SC_k} to obtain $I \parallel F \parallel SRV \parallel TS \parallel SI \parallel Sig_{SRV_k}$, then checks TS and Sig_{SRV_k} to verify its validity. After that, RC performs the following operation to determine whether the SRV is within C_{RC}:

$$\begin{aligned} J &= s'^{-1} \cdot F \bmod p' \\ &= s'^{-1} \cdot s' \cdot r_5[\alpha'^2(x_{CP} \cdot x_{SV} + y_{CP} \cdot y_{SV}) \\ &\quad + \alpha(x_{SV} \cdot d_1 + y_{SV} \cdot d_2) + r_3 \cdot d_3 + r_4 \cdot d_4] \bmod p' \end{aligned}$$

$$J' = \frac{J - (J \bmod \alpha'^2)}{\alpha'^2} = r_5(x_{SV} \cdot x_{SV} + y_{CP} \cdot y_{SV})$$

$$\begin{aligned} K &= F - 2J' \\ &= r_5[x_{CP}^2 + y_{CP}^2 + x_{SV}^2 + y_{SV}^2 - 2(x_{CP} \cdot x_{SV} \\ &\quad + y_{SV} \cdot y_{SV}) - R^2] \\ &= r_5[(x_{SV} - x_{CP})^2 + (y_{SV} - y_{CP})^2 - R^2]. \end{aligned}$$

Obviously, when $K \leq 0$, RC can determine that the RSV is within the circle query range C_{RC}, otherwise, the SRV is outside C_{RC}. Finally, RC can search all SRVs in C_{RC} with above operations, and we use CRVs to present these vehicles.

5) Ride-Hailing Request Generation: After RC obtains CRVs, which represents the vehicles around the RC's desired pick-up point, she/he can broadcast the ride-hailing request to CRVs through RS. Firstly, RC chooses a take-off point (x_{CT}, y_{CT}) on the map, and aggregates (x_{CT}, y_{CT}) and the encrypted quadtree EN to generate C_5 through executing FSSQ algorithm. Then, RC creates the signature $Sig''_{RC_k} = H(CRVs \parallel C_5 \parallel RC \parallel TS \parallel SI)^{SK_{RC_k}}$, and sends $E_{sk_{SC_k}}(CRVs \parallel C_5 \parallel RC \parallel TS \parallel SI \parallel Sig''_{RC_k})$ to RS.

6) Ride-Hailing Request Transmission: RS first decrypts $E_{sk_{SC_k}}(CRVs \parallel C_5 \parallel RC \parallel TS \parallel SI \parallel Sig''_{RC_k})$, and checks its validity. Then, RS runs FSSQ with C_5 to obtain the subregion where the RC's take-off point locates. After this, RS chooses a point ATP in this subregion randomly. Finally, RS makes the signature $Sig''_{RS} = H(ATP \parallel RS \parallel TS \parallel SI)^{SK_{RS}}$, and broadcasts $E_{sk_{CRVs}}(ATP \parallel RS \parallel TS \parallel SI \parallel Sig''_{RS})$ to CRVs, where sk_{CRVs} represent the session keys between RS and CRVs.

7) Make Decision and Ride-Hailing Response: Upon receiving $E_{sk_{CRVs}}(ATP \parallel RS \parallel TS \parallel SI \parallel Sig''_{RS})$, each CRV decrypts it with sk_{CRV_k} to obtain $ATP \parallel RS \parallel TS \parallel SI \parallel Sig''_{RS}$, and checks its validity. As a result, each CRV relies on RC's approximate take-off point ATP to make the decision of whether accepting this task. The accepting CRVs then send ”Accept Response” to RS. Finally, RS will return a ”Available CRVs List” to RC. After this, RC chooses one suitable CRV from ”Available CRVs List”, and tells this CRV through RS.

After that, RC and the CRV negotiate the session key $sk_{HC} = g^{SK_{RC_k} \cdot SK_{CRV_k}}$. Then, RC sends $E_{sk_{HC}}(P_{CP} \parallel PI \parallel Sig'''_{RC_k})$ to this CRV through RS, where P_{CP} is the coordinate of pick-up point (x_{CP}, y_{CP}), PI is personal information of RC, i.e., phone number, reputation, etc., and $Sig'''_{RC_k} = H(p_{CP} \parallel PI)^{SK_{RC_k}}$. Finally, the CRV verifies the packet's validity, and shares the real-time location on the map with RC.

Algorithm 2: ESVQ: Efficient and Secure Vehicle Query.

```
Input: A circle $C_{RC}$ with center $(x_{CP}, y_{CP})$ and radius $R$;
    Locations of RVs.
Output: The RV in $C_{RC}$ (CRV).
 1: RC generates a square $S_{RC}$ covering $C_{RC}$ with side $2R$.
 2: RC takes the vertexes of $S_{RC}$ as input, and executes
    FSSQ with RS.
 3: RS obtains a subregion covering $C_{RC}$ and SRVs.
 4: function decideSRV        ▷ Whether SRV is within $C_{RC}$
 5:     RC computes $D_i$.
 6:     SRV computes $F, I$.
 7:     RC computes $K$.
 8:     if $K > 0$ then
 9:         return $false$;       ▷ SRV is outside region $C_{RC}$
10:     else
11:         return SRV;           ▷ SRV is inside region $C_{RC}$
12:     end if
13: end function
```

Correctness of the ESVQ algorithm: We have approved the correctness of FSSQ above, and the correctness of ESVQ can be verified in the following. Based on the relationship of k_1, k_2, k_3, and k_4 presented before, it is obvious that FSSQ satisfied the constraints $r_5[\alpha'^2(x_{CP} \cdot x_{SV} + y_{CP} \cdot y_{SV}) + \alpha'(x_{SV} \cdot d_1 + y_{SV} \cdot d_2) + (r_3 \cdot d_3 + r_4 \cdot d_4)] < p'$ and $r_5[\alpha'(x_{SV} \cdot d_1 + y_{SV} \cdot d_2) + (r_3 \cdot d_3 + r_4 \cdot d_4)] < \alpha'^2$. Since the values of coordinates are not very big, we can choose applicable security parameters easily (such as $k_1 = 512$, $k_2 = 160$, $k_3 = 75$ and $k_4 = 75$). Note that the expression $K = r_5[(x_{RV} - x_{CP})^2 + (y_{RV} - y_{CP})^2 - R^2]$, where $(x_{RV} - x_{CP})^2 + (y_{RV} - y_{CP})^2 - R^2$ presents the distance between SRV and the center of the C_{RC} selected by RC, and r_5 is a positive random number. Therefore, whether SRV is within C_{RC} can be determined by the symbol of K.

V. Security Analysis

In this section, we analyze the security of the proposed TRACE. Specifically, following the security requirements discussed earlier, our analysis focuses on how to preserve the privacy of users' location information, the confidentiality of service provider's data, and the authentication during the ride-hailing process.

A. The Privacy of User's Sensitive Information Is Achieved

In our TRACE, users' sensitive information consists of three parts: a) (x_V, y_V), the location coordinate of RV, b) C_{RC}, the circular search region around the pick-up point of RC, and c) (x_{CT}, y_{CT}), the take-off point of RC.

Firstly, we approve that RS cannot obtain these sensitive information. Concretely, during the ride-hailing service, (x_V, y_V) and (x_{CT}, y_{CT}) are calculated with the encrypted quadtree, and masked before sending to RS's for location detection. In addition, C_{RC} is transformed into a square region S_{RC}, and the vertexes of S_{RC} $(x_{S1}, y_{S1}), (x_{S2}, y_{S2}), (x_{S3}, y_{S3}), (x_{S4}, y_{S4})$ are executed the same operations as (x_V, y_V) and (x_{CT}, y_{CT}) before sending to RS. We take (x_V, y_V) as an example, RV computes $A_i = A_{i1} \parallel A_{i2} \parallel A_{i3} \parallel A_{i4}$, where $A_{ij} = A_{ij1} \parallel A_{ij2}$, and $A_{ij1} = r_{ij} \cdot \alpha(x_V \cdot EN_{ij4} + y_V \cdot EN_{ij1} + EN_{ij6}) \bmod p$, $A_{ij2} = r_{ij} \cdot \alpha(x_V \cdot EN_{ij2} + y_V \cdot EN_{ij3} + EN_{ij5}) \bmod p$. Since RV keeps random number r_{ij} secret, (x_V, y_V) cannot be obtained by RS. Moreover, RV runs the random permutation function $\pi(A_{ij})$ to make the order of j chaotic, thus RS cannot infer the location relation between (x_V, y_V) and any edge of the subregion presented by N_i. Similarly, $(x_{CT}, y_{CT}), (x_{S1}, y_{S1}), (x_{S2}, y_{S2}), (x_{S3}, y_{S3})$, and (x_{S4}, y_{S4}) are also kept secret from RS. Therefore, RS cannot achieve any accurate location information of RC and RV, but from the subregion query results B_{ij}, RS can achieve the necessary commercial operating information, such as the distribution of ride-hailing behaviors and the density of RVs.

Secondly, we approve that RC and RV cannot obtain each other's sensitive information until a ride-hailing task is completed. During the vehicle query process, RC queries the RVs in the circle range C_{RC}, which is constructed with center (x_{CP}, y_{CP}) and radius R. Before sending to RV, RC masks (x_{CP}, y_{CP}) with secret random numbers s' and d_i to obtain $D = D_1 \parallel D_2 \parallel D_3 \parallel D_4$, where $D_1 = s'(x_{CP} \cdot \alpha' + d_1) \bmod p', D_2 = s'(y_{CP} \cdot \alpha' + d_2) \bmod p', D_3 = s' \cdot d_3 \bmod p'$, and $D_4 = s' \cdot d_4 \bmod p'$. Since s' and d_i are kept secret by RC, it is impossible for RV to obtain the (x_{CP}, y_{CP}). Moreover, RC sends $E = x_{CP}^2 + y_{CP}^2 - R^2$ to RV, obviously, RV cannot obtain the query radius R from E. Furthermore, RV computes the vehicles query response $F = s' \cdot r_5[\alpha'^2(x_{CP} \cdot x_{SV} + y_{CP} \cdot y_{SV}) + \alpha(x_{SV} \cdot d_1 + y_{SV} \cdot d_2) + r_3 \cdot d_3 + r_4 \cdot d_4] \bmod p'$ and $I = r_5(x_{SV}^2 + y_{SV}^2 + E)$. Due to the secret random numbers r_i are only known by RV, its accurate location coordinate (x_V, y_V) cannot be obtained by RC. Meanwhile, the vehicles query response F also contains the secret random numbers s' and d_i of RC, which achieve that only RC can recover the the vehicles query result K. In addition. F_3 and F_4 ensure that at least two random numbers are included in F, which can prevent the exhaustive attack of RC, and the radius R is 1 km, minimum, which guarantee that QU cannot infer the accurate location of RV through choosing a small circle search range. Meanwhile, attackers cannot achieve useful information even if they can capture users' data.

B. Confidentiality of Service Provider's Space Division Information

In order to analyze the distribution of ride-hailing behaviors and density of RVs on the map, RS divides the ride-hailing space into subregions, which is represented by a quadtree $N = \{N_1, N_2, \ldots, N_m\}$, where m is the number of quadtree nodes. Before sending to users, RS encrypts the original vertex data stored in each node of quadtree with the secret random numbers s and a_j. Specifically, for each node N_i, RS computes $EN_{ij1} = s(x_{Nij} \cdot \alpha + a_{j1}) \bmod p, EN_{ij2} = s(y_{Nij} \cdot \alpha + a_{j2}) \bmod p, \ldots, EN_{ij6} = s(x_{Nij'} \cdot y_{Nij} \cdot \alpha + a_{j6}) \bmod p$ to obtain $EN_{ij} = EN_{ij1} \parallel EN_{ij2} \parallel EN_{ij3} \parallel EN_{ij4} \parallel EN_{ij5} \parallel EN_{ij6}$ and $EN_i = EN_{i1} \parallel EN_{i2} \parallel EN_{i3} \parallel EN_{i4}$. It can be clearly seen that without knowing the random numbers s and a_{ji}, the original vertex data of each node is impossible to be retrieved. That is, it is impossible for RCs and RVs to obtain the space division information of RS. Moreover, the encrypted vehicle's location $A = A_{ij1} \parallel A_{ij2}$, where $A_{ij1} = r_{ij} \cdot \alpha(x_V \cdot EN_{ij4} + y_V \cdot EN_{ij1} + EN_{ij6}) \bmod p$ and $A_{ij2} = r_{ij} \cdot \alpha(x_V \cdot EN_{ij2} + y_V \cdot EN_{ij3} + EN_{ij5}) \bmod p$ also contains the secret random numbers s and a_j, therefore, only RC can recover the vehicles subregion query result B_{ij}. In addition, the existence of random numbers a_{ji} enhance the space of quadtree data, which can resist the exhaustive attack. Furthermore, Since the quadtree is encrypted, and transmitted through a secure channel, attackers cannot achieve the space division of RS. As a result, the confidentiality of space division information is ensured.

C. The Authentication of Data Packets Is Achieved

In TRACE, the packets flowing among RC, RV and RS are signed by BLS short signature [16]. Since the BLS short signature is provably secure under the computational Diffie-Hellman problem [20] in the random oracle model, the source authentication can be guaranteed. Moreover, for any unregistered user, since she/he does not have the secret key, she/he cannot send valid packets to RS or RV. As a result, if an unregistered user sends forged packets in the system, the malicious behavior can be detected in the proposed TRACE.

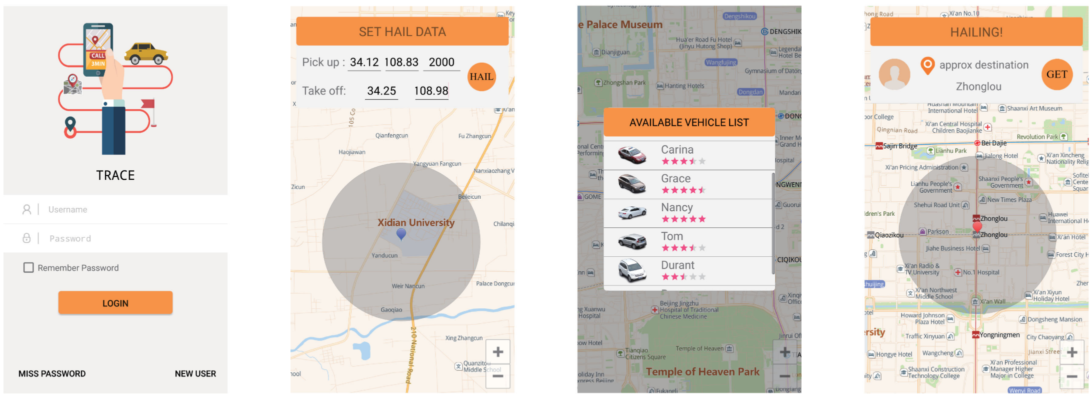
From the given analysis, we can conclude that the TRACE is secure and privacy-preserving for the users as well as RS, and can achieve our security goal.

VI. Performance Evaluation

In this section, we first evaluate the performance of the proposed TRACE in terms of the computation complexity of RC, RV, and RS. Then, we implement TRACE and deploy it in a real environment to evaluate its integrated performance.

A. Evaluation Environment

In order to measure the integrated performance, we implement TRACE on smartphone and PC with a real LBS dataset. Specifically, two smartphones with 2.2 GHz eight-core processor, 6 GB RAM, Android 7.1.1, and a PC with 2.0 GHz 6-core processor, 64 GB RAM, Windows 10, are chosen to evaluate RC,



(a) Register and login.

(b) RC sets ride-hailing information.

(c) RC obtains the available vehicles.

(d) RV receives the hailing requests for RV.

Fig. 6. Implementation of TRACE.

TABLE II
COMPUTATION COMPLEXITY IN SUBREGION QUERY

	RC/RV	RS (encrypt quadtree)	RS (analyze query results)
TRACE	$32M * t_{mul}$	$56M * t_{mul}$	$16(N + 5N') * t_{mul}$
PDCP	$(57 + 16l + 4l^2)M * t_{mul} + (25 + 16l)M * t_{exp}$	$24M * t_{mul} + 12M * t_{exp}$	$(14N + 70N' + 16l * N + 80l * N') * t_{exp} + (8 + 28N + 140N' + 4l * N + 20l * N')t_{mul}$

RV, and RS, respectively, which are connected through 802.11g WLAN. Based on our proposed TRACE, an application built in Java, named TRACE.apk, is installed in smartphones to simulate RC and RV, the simulator of RS is deployed in the PC. As shown in Fig. 6, users can register in RS, and access ride-hailing service. Specifically, Fig. 6(b) and (c) show that RC can choose the pick-up point on the map, set the query radius, and send ride-hailing request to RVs around the pick-up point. Then the RC can obtain the available vehicle list. In addition, RV can receive the hailing requests for the neighboring RCs, and obtain the RC's approximate destination, which is shown in Fig. 6(d).

B. Computation Complexity

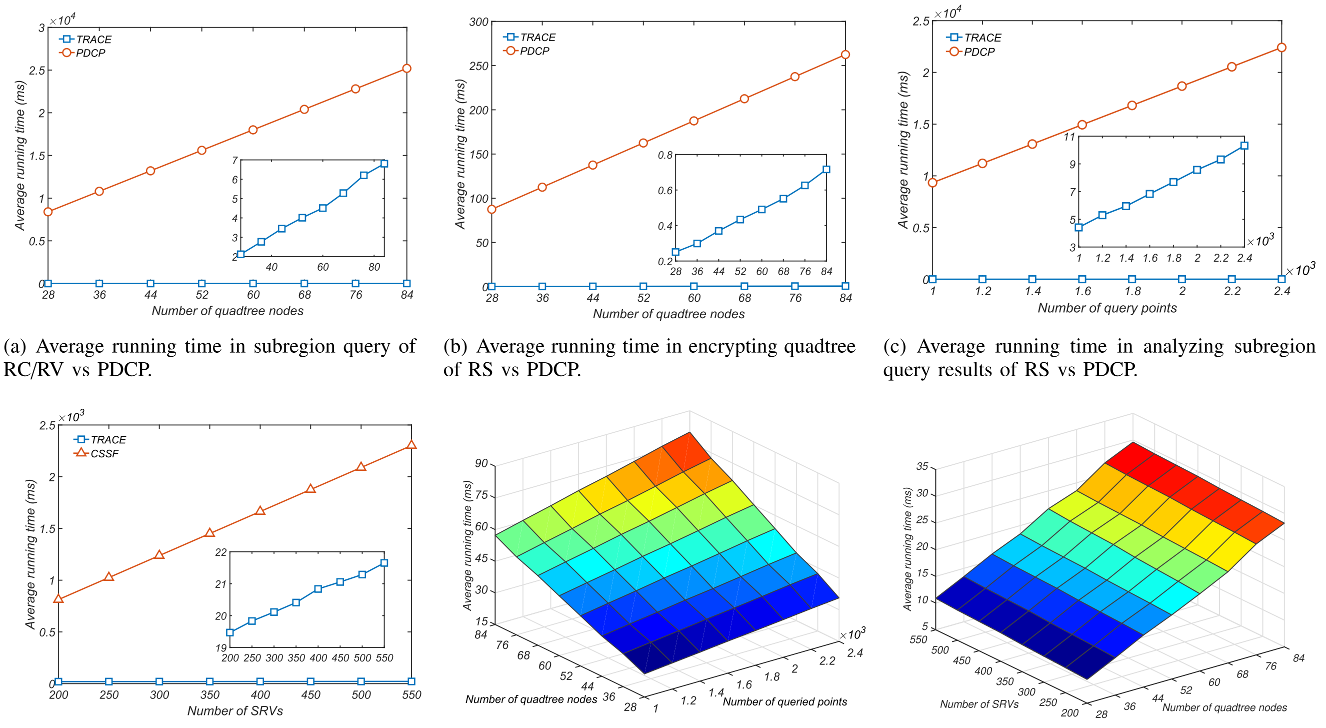
The TRACE is mainly constituted by two algorithms, FSSQ and ESVQ, which are constructed for dynamic subregion query of RS, and vehicles query of RC. In the following, we evaluate the computation complexity of RC, RV, and RS in the two processes separately.

During dynamic subregion query process, assume that the number of quadtree nodes, RV, and the requesting RC are M, N, and N', respectively. When encrypting the quadtree, RS requires $56M$ multiplication operations, and it costs $16(N + 5N')$ multiplication operations for RS to analyze the subregion query results of each node. And while blurring the coordinates of RC's take-off point and RV's location with the encrypted quadtree and random numbers, each RC and RV requires $32M$ multiplication operations. Denote that the time complexity for one multiplication as t_{mul}, Therefore, the computation complexity of RC/RV, encrypting the quadtree of RS, and analyzing the detection results of RS are $32M * t_{mul}$, $56M * t_{mul}$, and $16(N + 5N') * t_{mul}$, respectively.

In the process of vehicles query, RC first executes FSSQ for obtaining SRVs, which costs $128M$ multiplication operations of RC. Assume that the number of SRVs is L. Then, when masking the circle querying range, RC takes 6 multiplication operations. After receiving the search request from RC, each SRV takes 8 multiplication operations in aggregated computation. And it costs $2 + 5L$ multiplication operations for RC to analyze the query results. Therefore, the total computation complexity of RC, and SRV are $(8 + 5L + 128M) * t_{mul}$, and $8 * t_{mul}$, respectively.

Different from other time-consumption homomorphic encryption techniques, our proposed FSSQ and ESVQ algorithm use lightweight multiparity random masking technique, it can provide accurate subregion and vehicle query results and largely reduce the calculation times. In the following, for the comparison with TRACE, we select PDCP [21] and CSSF, which achieve the arbitrary convex polygon query, and circle range query with $Paillier$ [22] and $ElGamal$ [23] cryptosystem. Assume that the average domain size of quadtree nodes is measured by l and the time complexity for one exponentiation is presented by t_{exp}, then, one the one hand, during the subregion query process, the computation complexities for PDCP of RC/RV, encrypting the quadtree of RS, and analyzing the query results of RS are $(57 + 16l + 4l^2)M * t_{mul} + (25 + 16l)M * t_{exp}$, $24M * t_{mul} + 12M * t_{exp}$, and $(14N + 70N' + 16l * N + 80l * N') * t_{exp} + (8 + 28N + 140N' + 4l * N + 20l * N')t_{mul}$, respectively. On the other hand, in the vehicle query process, the computation complexities for CSSF of RC and RV are $(8 + L) * t_{exp} + (8 + 4L + 128M) * t_{mul}$, and $12 * t_{mul} + 4 * t_{exp}$, respectively.

Table II and Table III present the comparison of TRACE and PDCP/CSSF. We can clearly see that our proposed TRACE can



(a) Average running time in subregion query of RC/RV vs PDCP.

(b) Average running time in encrypting quadtree of RS vs PDCP.

(c) Average running time in analyzing subregion query results of RS vs PDCP.

(d) Average running time in vehicle query of RC vs CSSF.

(e) Integrated average running time in subregion query of RS.

(f) Integrated average running time in vehicle query of RC.

Fig. 7. Performance evaluation of computation complexity.

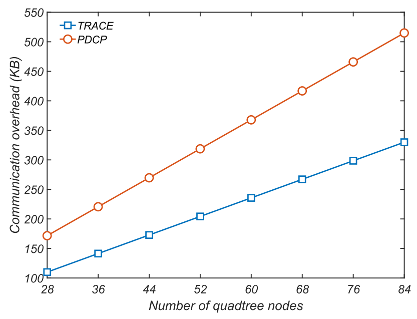
TABLE III
COMPUTATION COMPLEXITY IN VEHICLES QUERY

	RC	RV
TRACE	$(8 + 5L + 128M) * t_{mul}$	$8 * t_{mul}$
CSSF	$(8+L)*t_{exp}+(8+4L+128M)*t_{mul}$	$12 * t_{mul} + 4 * t_{exp}$

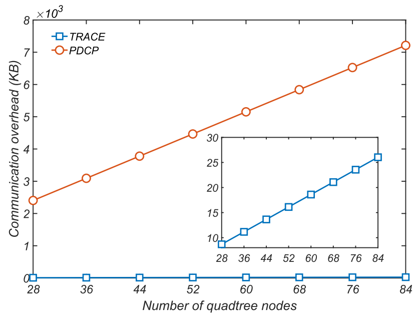
achieve privacy-preserving ride-hailing service with low complexity. The factor impacting the computation overhead of RC, RV, and RS during the subregion query process is the number of quadtree nodes and queried points, in Fig. 7(a), (b), and (c), we select the number of quadtree nodes from 28 to 84, and the number of queried points from 1000 to 2400, then we plot the computation overhead of RC/RV, encrypt the quadtree of RS, and analyzing the subregion query results of RS, respectively. From the figures, it can be obviously realized that with the increasing of quadtree nodes, the computation overhead of PDCP increases greatly, which is much higher than TRACE. On the other hand, the number of SRVs influences the computation overhead of RC during the vehicles query process, in Fig. 7(d), we choose the the number of SRVs from 200 to 550, and let the number of quadtree nodes is 60, the curve shows that the computation overhead of CSSF significantly increases and it is much higher than our TRACE. Although the computation overhead of TRACE also increases when then number of SRVs is large, it is still much lower than that of CSSF. In addition, in Fig. 7(e), we further plot the integrated computation overhead of RS varying with the numbers of quadtree nodes from 28 to 84, and detected points from 1000 to 2400, it can be seen that the computation overhead is still acceptable even if the number of users is large. Moreover, in Fig. 7(f), we plot the integrated computation overhead of RC varying with the increasing numbers of quedtree nodes from 28 to 84, and SRVs from 200 to 550, it can be clearly seen that the vehicle query time is available for handheld devices. As a result, the above analysis of computation complexity is verified via these figures, and our proposed TRACE can achieve better efficiency in terms of computation overhead in RC, RV, and RS.

C. Communication Overhead

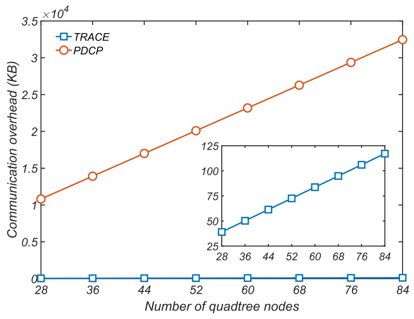
In TRACE, during the subregion query process, RS first broadcasts the encrypted quadtree $E_{sk_{users}}(\alpha \parallel p \parallel EN \parallel RS \parallel TS \parallel SI \parallel Sig_{RS})$ to users, then, RV will submit the subregion query response $E_{sk_{SV_k}}(A \parallel RV \parallel TS \parallel SI \parallel Sig_{RV_k})$ to RS. Moreover, RC sends her/his encrypted vehicle query request $E_{sk_{SC_k}}(C \parallel VSQ \parallel RC \parallel TS \parallel SI \parallel Sig'_{RC_k})$, and encrypted ride-hailing request $E_{sk_{SC_k}}(CRVs \parallel C_5 \parallel RC \parallel TS \parallel SI \parallel Sig_{RC_k})$ to RS. In the process of vehicles query, the data transmitted between RC and RV is the vehicles query packet VSQ, and the vehicle query response $E_{sk_{SC_k}}(I \parallel F \parallel SRV \parallel TS \parallel SI \parallel Sig_{SRV_k})$ of RV, respectively. In the real environment, we record the size of these packets, and compare with PDCP/CSSF in one round. Similar to computation complexity, the factor impacting the computation overhead among RC, RV, and RS is the number of quadtree nodes and queried points, therefore, in Fig. 8(a), (b), and (c) we select the number of quadtree nodes from 28 to 84, and the number of queried points from 1000 to 2400, then we plot the computation



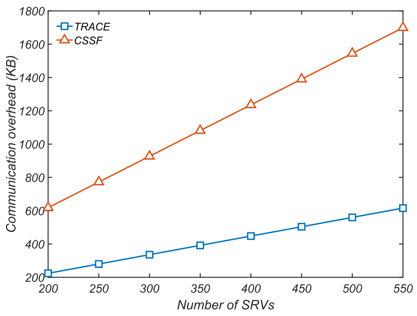
(a) Communication overhead of encrypted quadtree vs PDCP.



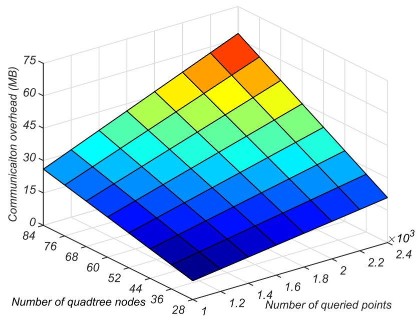
(b) Communication overhead between RS and RV vs PDCP.



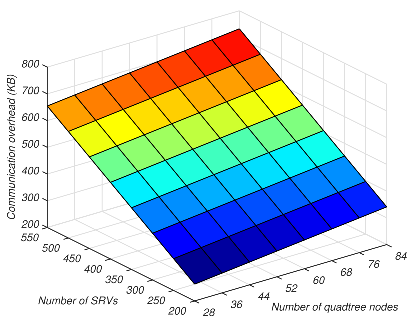
(c) Communication overhead between RS and RC vs PDCP.



(d) Communication overhead between RC and RV vs CSSF.



(e) Integrated communication overhead of RS.



(f) Integrated communication overhead of RC.

Fig. 8. Performance evaluation of communication overhead.

overhead among RC, RV, and RS, the figures show that with the increasing number of quadtree nodes, the communication overhead of PDCP significantly increases and it is much higher than that of TRACE. On the other hand, the number of SRVs influences the communication overhead of RC, in Fig. 8(d), we choose the the number of SRVs from 200 to 550, and let the number of quadtree nodes is 60, the curve shows that the computation overhead of CSSF significantly increases and it is also much higher than our proposed TRACE. In addition, in Fig. 8(e), we plot the integrated communication overhead of RS varying with the numbers of quadtree nodes from 28 to 84 and detected points from 1000 to 2400, the figure shows that the communication overhead is acceptable for RS even if the number of users is large. Furthermore, Fig. 8(f) shows the integrated communication overhead of RC varying with the number of SRVs from 200 to 550, and the number of quadtree nodes from 28 to 84, it can be clearly seen that RC can maintain a low communication overhead. In conclusion, our proposed TRACE can achieve better efficiency in terms of communication overhead in RC, RV, and RS.

VII. Related Work

The location privacy has gained great interest in recent years, in this section, we briefly discuss some of them closely related to ours.

Many previous works on location privacy-preserving is based on anonymization techniques, which prevent the user's accurate location data via blurring the users' location into a cloaked spatial region. Cui *et al.* [24] proposed a novel hierarchical Hilbert Curve spatial cloaking algorithm to effectively achieve k-anonymity for mobile users in LBS, the proposed scheme considers the Average Query Density (AQD), and generates the Anonymity Set (AS) satisfying reciprocity and uniformity, which guarantees high privacy level. Jadallah *et al.* [25] presented an efficient k-anonymity algorithm to compute the cloaked area with minimal number of communication rounds between the user and the cloud server, which reduces the search time by starting the search at an intermediate estimated level of the indexing structure that is as close as possible to the queried location. In addition, there are also some works combined the anonymization techniques with quadtree data structure. Gruteser and Grunwald [13] proposed a well-known location privacy-preserving approach, in this approach, a user's location will be reported as a two-dimensional spatial cloaking area where at least $k-1$ other users are also in the same area, meanwhile, a quadtree based cloaking algorithm has been designed, and the size of the anonymity set k is used to measure the degree of anonymity. Wang *et al.* [26] proposed quadtree based algorithms to generate cloaking areas satisfying users' privacy requirements, the proposed algorithms focus on finding the smallest cloaking area for each location request, and can be used in protecting the users' location privacy in real-life scenarios. Ju *et al.* [27] proposed a new location privacy protection scheme through constructing a quadtree entropy map to generate cloaking boxes, which enables users to protect their location privacy relying only on their smartphones while the trusted anonymization server is not required in the proposed scheme. However, due to the characteristic anonymization techniques, the above privacy framework commonly returns a list of candidate answers according to the cloaked users or areas, which brings heavy communication overhead.

Cryptographic techniques, such as homomorphic encryption and secure multi-party computation (SMC), are also widely used in location privacy-preserving. Yi *et al.* [28] proposed a solution for mobile users to preserve their locations and query privacy in approximate k-nearest neighbor, which is built on the $Paillier$ public-key cryptosystem, and can provide both location and query privacy. Mu and Bakiras [21] presented a novel approach which allows a mobile user to choose an arbitrary convex polygon on the map, and detect whether her/his friends are within the polygon, the proposed approach is based on $Paillier$ and $ElGamal$. Zhu *et al.* [29] constructed a scheme, with which the LBS provider can compute the centroid of a companion set while it does not know the locations of the members in the set, and the authors proposed a novel key agreement protocol to construct the perturbations which is used to disguise real locations. Moreover, there are also some works combined the cryptographic techniques with quadtree data structure. Aiming at protecting the location privacy of users in range and nearest neighbor queries, Khoshgozaran *et al.* [30] divided the query space with quadtree, and proposed a dual curve query resolution approach with $Helbert$ curve to improve the performance of blind range and KNN queries. Meanwhile, the authors designed space transformation employ space filling curves with cryptographic one-way hash functions, which can achieve stronger and privacy than that of retrieval-based approaches commonly used spatial cloaking techniques without incurring the prohibitive costs of private information. To enable the secure management of private user locations in non-trusted systems, Wernke *et al.* [15] presented two novel location sharing approaches based on the concept of multi-secret sharing and quadtree data structure, the proposed approaches are improved from the exiting geometric position sharing approaches by Durr *et al.* [31] and Skvortsov *et al.* [32]. However, some high time-consuming operations are required in the most above-mentioned schemes, which brings heavy computation overhead.

Different from above works, our proposed TRACE aims at efficiency and privacy issues, and based on quadtree data structure, lightweight multi-party random masking technique, and point in polygon strategy, we develop an efficient and privacy-preserving dynamic spatial query scheme for ride-hailing service. In particular, our proposed TRACE can provide precise spatial query while protecting users' accurate location data as well as ensuring the confidentiality of service provider's space division information, and can be easily implemented in smart phones and computers due to its high efficiency.

VIII. Conclusion

In this paper, we proposed an efficient and privacy-preserving dynamic spatial query scheme for online ride-hailing service, called TRACE. Based on random masking technique and point in polygon strategy, in TRACE, users can access ride-hailing service without leaking their accurate location information, and the ride-hailing server can achieve the necessary commercial operating information while keeping its space division data confidential. Specifically, before being sent out, all of the location-related data were masked into chipertexts by its owner, and were calculated without decryption during the spatial query process. Therefore, RC, RV, and RS cannot obtain each other's sensitive information. Meanwhile, based on quadtree data structure, the proposed scheme improved the efficiency of dynamic spatial query greatly. Detailed security analysis showed its security strength and privacy-preserving ability, and extensive experiments were conducted to demonstrate its efficiency.

Availability

The implementation of the proposed scheme and relevant information can be downloaded at https://www.xdzhuhui.com/demo/TRACE/.

References

[1] D. N. Anderson, "Not just a taxi? For-profit ridesharing, driver strategies, and VMT," *Transportation*, vol. 41, no. 5, pp. 1099–1117, 2014.

[2] E. Uhlemann, "Its frequency bands are being debated [connected vehicles]," *IEEE Veh. Technol. Mag.*, vol. 11, no. 4, pp. 12–14, Dec. 2016.

[3] W. Zhao, Y. Qin, D. Yang, L. Zhang, and W. Zhu, "Social group architecture based distributed ride-sharing service in VANET," *Int. J. Distrib. Sensor Netw.*, vol. 2014, no. 1, pp. 1–8, 2014.

[4] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," *Proc. Nat. Acad. Sci. USA*, vol. 111, no. 37, pp. 13290–13294, 2014.

[5] N. Fei, Y. Zhuang, J. Gu, J. Cao, and L. Yang, "Privacy-preserving relative location based services for mobile users," *China Commun.*, vol. 12, no. 5, pp. 152–161, 2015.

[6] C. Huang, Z. Yan, N. Li, and M. Wang, "Secure pervasive social communications based on trust in a distributed way," *IEEE Access*, vol. 4, pp. 9225–9238, 2016.

[7] H. Zhu, X. Liu, R. Lu, and H. Li, "Efficient and privacy-preserving online medical prediagnosis framework using nonlinear SVM," *IEEE J. Biomed. Health Informat.*, vol. 21, no. 3, pp. 838–850, May 2017.

[8] B. Wang, M. Li, H. Wang, and H. Li, "Circular range search on encrypted spatial data," in *Proc. IEEE Conf. Commun. Netw. Secur.*, 2015, pp. 182–190.

[9] H. Zhu, F. Wang, R. Lu, F. Liu, G. Fu, and H. Li, "Efficient and privacy-preserving proximity detection schemes for social applications," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2947–2957, Aug. 2018.

[10] H. Zhu, R. Lu, C. Huang, L. Chen, and H. Li, "An efficient privacy-preserving location based services query scheme in outsourced cloud," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7729–7739, Sep. 2016.

[11] C. Huang, R. Lu, and H. Zhu, "Privacy-friendly spatial crowdsourcing in vehicular networks," *J. Commun. Inf. Netw.*, vol. 2, no. 2, pp. 59–74, 2017.

[12] L. Sweeney, "k-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl. Based Syst.*, vol. 10, no. 5, pp. 557–570, 2002.

[13] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. Int. Conf. Mobile Syst., Appl., Serv.*, 2003, pp. 31–42.

[14] M. Ashouri-Talouki and A. Baraani-Dastjerdi, "Homomorphic encryption to preserve location privacy," *Int. J. Secur. Appl.*, vol. 6, no. 4, pp. 183–190, 2012.

[15] M. Wernke, F. Dürr, and K. Rothermel, "PShare: Ensuring location privacy in non-trusted systems through multi-secret sharing," *Pervasive Mobile Comput.*, vol. 9, no. 3, pp. 339–352, 2013.

[16] B. Dan, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2001, pp. 514–532.

[17] W. G. Aref and H. Samet, "Efficient window block retrieval in quadtree-based spatial databases," *Geoinformatica*, vol. 1, no. 1, pp. 59–91, 1997.

[18] R. Lu, H. Zhu, X. Liu, J. K. Liu, and J. Shao, "Toward efficient and privacy-preserving computing in big data era," *IEEE Netw.*, vol. 28, no. 4, pp. 46–50, Jul./Aug. 2014.

[19] F. Feito, J. C. Torres, and A. Urena, "Orientation, simplicity, and inclusion test for planar polygons," *Comput. Graph.*, vol. 19, no. 4, pp. 595–600, 1995.

[20] F. Zhang and P. Wang, “On relationship of computational Diffie-Hellman problem and computational square-root exponent problem,” in *Proc. Int. Conf. Coding Cryptol.*, 2011, pp. 283–293.

[21] B. Mu and S. Bakiras, “Private proximity detection for convex polygons,” in *Proc. Int. ACM Workshop Data Eng. Wireless Mobile Access*, 2013, pp. 36–43.

[22] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 1999, pp. 223–238.

[23] T. Elgamal, “A public key cryptosystem and a signature scheme based on discrete logarithms,” *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, 1985.

[24] N. Cui, X. Yang, and B. Wang, “A novel spatial cloaking scheme using hierarchical hilbert curve for location-based services,” in *Proc. Int. Conf. Web-Age Inf. Manage.*, 2016, pp. 15–27.

[25] H. Jadallah and Z. A. Aghbari, “Aman: Spatial cloaking for privacy-aware location-based queries in the cloud,” in *Proc. Int. Conf. Internet Things Cloud Comput.*, 2016, pp. 1–6.

[26] Y. Wang, D. Xu, X. He, C. Zhang, F. Li, and B. Xu, “L2p2: Location-aware location privacy protection for location-based services,” in *Proc. IEEE Int. Conf. Comput. Commun.*, 2012, pp. 1996–2004.

[27] X. Ju and K. G. Shin, “Location privacy protection for smartphone users using quadtree entropy maps,” *J. Inf. Privacy Secur.*, vol. 11, no. 2, pp. 62–79, 2015.

[28] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, “Practical approximate k nearest neighbor queries with location and query privacy,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 6, pp. 1546–1559, Jun. 2016.

[29] X. Zhu, Y. Lu, X. Zhu, and S. Qiu, “A location privacy-preserving protocol based on homomorphic encryption and key agreement,” in *Proc. Int. Conf. Inf. Sci. Cloud Comput. Companion*, 2014, pp. 54–59.

[30] A. Khoshgozaran, H. Shirani-Mehr, and C. Shahabi, “Blind evaluation of location based queries using space transformation to preserve location privacy,” *GeoInformatica*, vol. 17, no. 4, pp. 599–634, 2013.

[31] F. Durr, P. Skvortsov, and K. Rothermel, “Position sharing for location privacy in non-trusted systems,” in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2011, pp. 189–196.

[32] P. Skvortsov, F. Dürr, and K. Rothermel, “Map-aware position sharing for location privacy in non-trusted systems,” in *Proc. Int. Conf. Pervasive Comput.*, 2012, pp. 388–405.

Fengwei Wang (S’18) received the B.Sc. degree from Xidian University, Xi’an, China, in 2016. He is currently working toward the Ph.D. degree at the School of Cyber Engineering, Xidian University, Xi’an, China.

His research interests include applied cryptography, cyber security and privacy.

Hui Zhu (M’13) received the B.Sc. degree from Xidian University, Xi’an, China, in 2003, the M.Sc. degree from Wuhan University, Wuhan, China, in 2005, and the Ph.D. degree from Xidian University, in 2009.

In 2013, he was with the School of Electrical and Electronics Engineering, Nanyang Technological University as a Research Fellow. Since 2016, he has been a Professor with the School of Cyber Engineering, Xidian University. His research interests include applied cryptography, cloud security, and big data security and privacy.

Ximeng Liu (S’13–M’16) received the B.Sc. degree in electronic engineering from Xidian University, Xi’an, China, in 2010 and the Ph.D. degree in cryptography from Xidian University, in 2015.

He is currently a Full Professor with College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. Also, he is a Research Fellow with the School of Information System, Singapore Management University, Singapore. His research interests include cloud security, applied cryptography, and big data security.

Rongxing Lu (S’09–M’10–SM’15) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He has been an Assistant Professor with the Faculty of Computer Science, University of New Brunswick (UNB), Fredericton, NB, Canada, since August 2016. Before that, he was an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, from April 2013 to August 2016. He was a Postdoctoral Fellow with the University of Waterloo from May 2012 to April 2013. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He was the recipient of the the most prestigious “Governor Generals Gold Medal,” and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. He is currently a Senior Member of the IEEE Communications Society. He is currently the Secretary of the IEEE ComSocCIS-TC.

Fenghua Li received the B.S. degree in computer software, the M.S. and Ph.D. degrees in computer systems architecture from Xidian University, Xi’an, China, in 1987, 1990, and 2009, respectively.

He is currently a Professor and a Doctoral Supervisor with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. He is also a Doctoral Supervisor with the Xidian University. His research interests include network security, system security, privacy computing, and trusted computing.

Hui Li (M’10) received the B.Sc. degree from Fudan University, Shanghai, China, in 1990, and the M.Sc. and Ph.D. degrees from Xidian University, Xi’an, China, in 1993 and 1998, respectively.

Since 2005, he has been the Professor with the School of Telecommunication Engineering, Xidian University, Xi’an, China. His research interests include cryptography, wireless network security, information theory, and network coding. He was a TPC Co-Chair of ISPEC 2009 and IAS 2009, a General Co-Chair of E-Forensic 2010, ProvSec 2011, and ISC 2011, a Honorary Chair of NSS 2014 and ASIACCS 2016.

Songnian Zhang received the master’s degree from Xidian University, Xi’an, China, in 2016.

He is currently a Research Assistant with the School of Cyber Engineering, Xidian University, Xi’an, China. His research interests include cyber security, big data privacy, and IoT security.