ASAP: An Anonymous Smart-Parking and Payment Scheme in Vehicular Networks

Liehuang Zhu, *Member, IEEE*, Meng Li, *Student Member, IEEE*,
Zijian Zhang, *Member, IEEE*, and Zhan Qin, *Member, IEEE*

Abstract—Cruising for a vacant and economical parking spot causes not only time-consuming and frustrating driving experiences, but fuel waste and air pollution. Public parking spots in crowded cities are scarce and expensive. On the contrary, private parking spots usually have low utilization rates, and the spot suppliers are willing to provide their extra parking resources due to a maintenance cost by charging parking fees. Given this situation, it is imperative to call for a smart parking system that collects and provides private parking spots (e.g., around home or workplace) to ease public parking concerns. However, when the suppliers (drivers) are providing (querying for) parking spots, their privacy (e.g., location, identity) is inevitable to be disclosed and existing parking schemes cannot achieve anonymous authentication and anonymous payment simultaneously. To tackle these problems, we propose an anonymous smart-parking and payment (ASAP) scheme in vehicular networks. Specifically, we use short randomizable signature to provide anonymity and conditional privacy. We achieve quick result matching with hashmap and anonymous payment with E-cash. Security analysis and experimental results show that ASAP can protect privacy in a conditional way and has low computational costs and communication overhead.

Index Terms—Vehicular networks, smart parking, security and privacy, anonymous payment

1 Introduction

The parking in downtown areas has been a problem for years [1]. Reports [2], [3] show that nearly 1.3 million out of 5.7 million motor vehicles regularly struggle to find parking places and by 2014 there were 2.9 million spaces were in need at night when Shanghai's 3 million registered car owners get off work. Meanwhile, a study [4] shows that there are 30 percent of the traffic congestion is caused by the drivers who are cruising for parking spots. Another record [4] shows that cars cruising for parking spots traveled 945,000 extra miles, burned 47,000 gallons of gasoline and produced 728 tons of carbon dioxide in a Los Angeles district for over a year. The situation is getting even worse in developing countries where the number of vehicles has been increasing without sufficient investment in parking facilities. Some governments try to mitigate these problems through building extra parking lots, deploying road-side sensors, and establishing parking guidance systems. While the effect of such centralized approach is obvious and immediate, the limited construction space, expensive investment, and the consequent maintenance cost inhibit a widespread adoption. Therefore, the parking problem cannot be solved efficiently only through public infrastructure or management.

Different from the traditional solutions, we have observed that a large proportion of parking spots is owned by the private sector and beyond the direct control of local transport authority. Such parking spots in private sector (e.g., residential space, workplace) always remain vacant when spot suppliers (we will use supplier for short) are on a trip or off duty. In addition, suppliers usually spend much money on buying and maintaining these private spots. Hence, they are willing to offer their parking spots for a parking fee as an economical compensation for their expenses. These motivate us to think how much time will be saved for cruising drivers and how much traffic congestion will be relieved and if the information of private parking spots can be initiatively provided by suppliers the public, especially the cruising drivers.

The recent increase in the development and use of smart phones has provided the opportunity to collaboratively sense and share information for the common good. Individuals with sensing, storing and computing devices [5], [6], [7] are now able to collect and contribute valuable data (usually in a form of a report) to a server for different applications, such as finding parking spots. Therefore, we can utilize the people to help improve smart-parking [8], [9], [10], [11], [12], [13], [14] with timely and accurate parking information.

However, security and privacy issues are preliminary concerns for users (including suppliers and drivers) participating in the data collecting and sharing task, since the system is faced with various cyber attacks and the private information of users is put at risk [15]. If the private parking spots are published online or a driver continues to upload information (e.g., current location, visit to a hospital, dinner

- *L. Zhu, M. Li, and Z. Zhang are with the School of Computer Science and Technology, Beijing Institute of Technology, Haidian Qu, Beijing Shi 100081, China. E-mail: {liehuangz, menglibit, zhangzijian}@bit.edu.cn.*
- *Z. Qin is with the Department of Electrical and Computer Engineering, University of Texas at San Antonio, San Antonio, TX 78249. E-mail: zhan.qin@utsa.edu.*

Manuscript received 15 July 2017; revised 26 May 2018; accepted 19 June 2018. Date of publication 26 June 2018; date of current version 9 July 2020. (Corresponding author: Zijian Zhang.)
Digital Object Identifier no. 10.1109/TDSC.2018.2850780

list in a restaurant) without any security protection, the untrustworthy server or an adversary can infer sensitive information such as home/work address, health condition, insurance status, salary level, diet preference and even identity by analyzing transmitting messages along with background knowledge. If these issues cannot be well addressed, the system's functionality and durability will be endangered. To prevent illegal suppliers or drivers from submitting invalid messages to the server, registration is necessary for all entities which will be authenticated in each report and query to make sure that they are the registers. Data confidentiality and integrity are also important security issues. A supplier does not want the server or people nearby to learn the spot status by eavesdropping on the reports and a driver does not want the server or people nearby to learn his cruising need for a parking spot. The messages sent to the server have to be signed to guarantee that they are not forged. Location privacy is another concern for users. Nevertheless, the server has to know a user' location in order to return the location-related result. Therefore, we use cloaking [16] to inhabit users' area with other locations and further satisfy anonymity constraint.

While providing private parking spots to unacquainted drivers, neighbors or colleagues of the parking spot supplier will have various concerns about the inconvenience or threat from an outsider who may have huge interest for their privacy. On the other hand, a maliciously behaved supplier may abuse the privacy-preserving mechanism by providing inaccurate or others' parking spots to gain profits. These potential situations particularly happen when there is a dispute or accident. Therefore, we should provide anonymous authentication to protect users' privacy as well as traceability [17] meaning a trusted authority can track a targeted supplier or driver.

After the server has collected enough supplying reports, it should provide an accurate and quick service [18] which in this case is the parking spot, to drivers. Here, we constructed a hashmap storing all parking locations with pertinent information and since we used binary tree in the hashmap, the average matching time is $O(log_2 Num)$ which is the same for deleting and inserting time, where Num is the number of users.

Since the parking spot is private, supplier will charge a parking fee for using the private resource but anonymous authentication makes it difficult to complete the transaction. In summary, the key challenges are: (1) we need to design a system to encourage the suppliers to willingly offer their parking spots to the public because different from public parking spots, private parking spots are hidden from the public; (2) we need to protect the drivers' and suppliers' privacy because if these private parking spots as well as drivers' queries are published online without any protection, an adversary with background knowledge will acquire the suppliers' and drivers' privacy, such as home/work address, health condition, insurance status, salary level, diet preference and even identity; (3) We need to achieve payment process after drivers finishing parking even when suppliers and drivers are anonymous to the server. Existing public parking schemes do not work in our scenario for several reasons: (1) they do not possess the functionality to utilize private parking spots; (2) they cannot enable spot suppliers to have requirements for vehicles to be parked in their spots, such as the parking duration and the size of the vehicle; (3) they cannot enable drivers to have preferences for private parking spots, such as location and price.

In this paper, we propose an Anonymous Smart-pArking and Payment (ASAP) scheme in vehicular networks to provide private parking spots for cruising drivers. ASAP consists of six phases: system initialization, driver querying, supplier reporting, result retrieving, identity disclosing, and anonymous payment. Specifically, we use short randomizable signature [19] to anonymously offer and query the private parking information to a server. Second, we construct a hashmap storing all the parking spot information in the cloud for quick matching and return the parking results sorted by the price. Last, we establish a connection between driver and supplier while utilizing E-cash [20] to solve the anonymous payment problem from driver to supplier.

The main contributions of this paper are as follows.

- We propose an novel anonymous smart-parking and payment scheme in vehicular networks. With the private parking spots provided from suppliers and processed by server, cruising drivers can find extra parking spots besides public lot with less fuel and time, suppliers will improve the utility of their spots and make a profit, and public traffic congestion is further reduced.
- ASAP preserves the users' privacy by utilizing the short randomizable group signature [19]. Specifically, a supplier (driver) sends a supplying report (parking query) to the server anonymously. Meanwhile, a trusted authority can trace a targeted user if a dispute happens.
- ASAP achieve quick parking result retrieving through hashmap storing all parking spot information and achieve anonymous payment with E-cash from driver to supplier even though anonymous authentication is adopted to protect the users' privacy.
- We run extensive experiments to evaluate the communication overhead and computational costs of ASAP, and formally prove the security and privacy of ASAP.

We review related work in Section 2. In Section 3, we formalize the system model, threat model and design objectives. Then, we revisit the preliminaries in Section 4. Section 5 presents our ASAP scheme, followed by security and privacy analysis in Section 6 and performance evaluation in Section 7, respectively. We also add some discussions in Section 8. Finally, we conclude this paper in Section 9.

2 Related Work

Existing work has been focusing on parking or parking related problem in VANET [8], [9], on-street parking [10], [11], and navigation [12], [13], [14].

Lu et al. [8] proposed a secure navigation scheme SPARK to provide drivers with accurate and convenient parking services in large parking lots, including real-time parking navigation service, intelligent antitheft protection, and friendly parking information dissemination. Lu et al. [9] presented a new intelligent and secure privacy-preserving parking scheme by employing parking lot RSUs to manage the whole parking lot. ParkNet [10] is a system that estimates street parking availability by using vehicles equipped with a GPS receiver and a passenger-side-facing ultrasonic rangefinder. The data is then aggregated at central server, building a real-time map of parking availability and providing this information to drivers in search of parking. Parksense [11] is another system that leverages the ubiquity of WiFi beacons to monitor on-street parking availability. It utilizes a robust Wi-Fi signature matching approach to detect a driver's return to the parked vehicle and it uses a novel approach based on the rate of change of Wi-Fi beacons to sense if the user has started driving.

Adjacent to parking, there are several navigation schemes [12], [13], [14] in the literature. Chim et al. [12] proposed a secure and privacy-preserving navigation system VSPN. They use anonymous credential to protect drivers' privacy, meaning that the driver who issues the query is guaranteed to be unlinkable to any party including the trusted authority. Ni et al. [13] proposed a privacy-preserving real-time navigation (PRIN) system. The RSUs cooperatively find an optimal path for a querying vehicle to the destination according to the real-time traffic information provided by the vehicles in their coverage areas. Meanwhile, the trusted authority can disclose the drivers' identities if they upload inaccurate traffic information. Ni et al. [14] proposed a cloud-based privacy-preserving parking navigation (CPARN) system through vehicular communications. The drivers are guided by a server to vacant parking spaces near their destinations without compromising privacy, including drivers' identities, references and routes. However, there are two significant differences between their schemes and ours. First, the previous smart parking work did not take private parking spots into consideration. Second, our scheme includes a payment phase.

There are also some works on membership revocation [21], [22]. Gisdakis et al. [21] proposed a participatory sensing architecture SPPEAR to protect user privacy and support user incentive mechanisms. They systematically discussed key aspects such as privacy, security, accountability and incentives provision. Particularly, SPPEAR supports pseudonym revocation to shun out offending users which is similar to our identity disclosing phase. However, the number of system entities (task, service, group manager, identity provider, pseudonym certification authority, sample aggregation service and resolution authority) in SPPEAR is more than that in ASAP, and it is not necessary for ASAP to expend system model since we mainly aim to provide smart parking and anonymous payment, and extra roles require extra deployment costs. Rahaman et al. [22] proposed a provably secure group signature scheme SRBE to support sublinear revocation through utilizing and integrating cryptographic, algorithmic, and data structural building blocks. Specifically, SRBE uses time bound pseudonyms as revocation tokens for fast revocation check.

Anonymous payment works include fair payment system (FPS) [23], WhoPay [24] and Bitcoin [25]. FSP [23] claimed that an anonymous payment systems could be misused by criminals for perfect blackmailing or money laundering because the payment anonymity prevented the bank from tracing money. Thereby, FSP proposed to use an additional trusted party, a judge, to remove the anonymity of a transaction. However, this is not the focus of our work and we can recover a misbehaved user's real identity as well. Meanwhile, the privacy of a shop is not guaranteed because the transfer is from a customer's anonymous account to the shop's account which is public, making it not applicable to ASAP. Moreover, the bank in FSP has to manage two types of accounts (i.e., personal accounts and anonymous accounts) which will double the storage costs and there are multiple interactions between a customer and the bank. WhoPay [24] is a payment system where users remain anonymous but a trusted authority can reveal relevant identities when it is necessary and it distributes a coin transfer load across all peers. However, even though the holdership of a coin can be transferred between users, the interactions between users will bring extra communication overhead, and it is not applicable to ASAP since we do not need a driver to issue a transfer proof to another driver. Bitcoin [25] removes the central authority which is a bank and constructs a public ledger to reliably store all transactions. Miners in the network continuously computes the SHA256 hash function to find a pre-image which hashes to an output under required format.

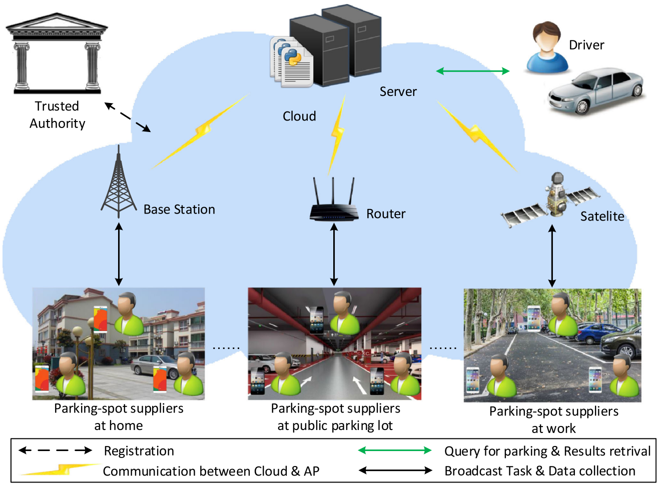

Trusted Authority
Cloud
Server
Driver
Base Station
Router
Satelite
......
......
Parking-spot suppliers at home
Parking-spot suppliers at public parking lot
Parking-spot suppliers at work
Registration
Communication between Cloud & AP
Query for parking & Results retrival
Broadcast Task & Data collection

Fig. 1. System architecture.

3 System Model

3.1 System Model

The system model mainly consists of four entities: trusted authority, server, driver, and supplier, as shown in Fig. 1. The key notations are listed in Table 1.

- *Trust Authority (TA)* is a powerful entity whose responsibility is initializing the whole system which includes registering drivers and suppliers, generating public parameters, and distributing keys. TA will be offline unless a dispute arises where it can trace a targeted user's identities.
- *Server* receives the supplying reports from suppliers and parking queries from drivers, then it searches the database and returns matching results to drivers. The server generates electric coupons to complete anonymous payment and help suppliers verify the validity of coupons.
- *Supplier* is willing to contribute her private parking spot to the driver and she will charge the driver a

TABLE 1
Key Notations

Notation	Definition
$(x, y), (\tilde{g}, \tilde{X}, \tilde{Y})$	TA's group secret key, group public key
ID_s, s_0, D_s	Identity, secret key, and public key
gsk_{ID_s}, gpk_{ID_s}	Group secret key, group public key
$Pid_i, Supp_i$	Pseduo-id and cloaked parking location of i
t_1, t_2, pr_i	Supplying time, expired time and spot price of i
k_i, K_i	i's Temporary secret key, temporary public key
C_{i1}, C_{i2}	Ciphertext of $K_i, Supp_i, pr_i$
$(S'_{i1}, S'_{i2}, c_i, ss_i)$	Signature on $(PID, Supp_i, t_1, t_2, K_i, pr_i)$
S'_{i3}	Ciphertext of S_{i3} for authentication
$r_1, r_2; Rep_i$	Randomize factor; i's supplying report
ID_d, d_0, D_d	Identity, secret key, and public key
$gsk_{ID_d},$	Group secret key, group public key
Pid_j, Pa_j	Pseduo-id and desired parking location of j
tt_1, tt_2	Querying time and expired time of j
k_j, K_j	j's temporary secret key, temporary public key
C_{j1}, C_{j2}	Ciphertext of K_j, Pa_j
$(S'_{j1}, S'_{j2}, c_j, ss_j)$	Signature on $(Pid_j, Pa_j, tt_1, tt_2, K_j)$
S'_{j3}	Ciphertext of S_{j3} for authentication
$r_3, r_4; Que_j$	Randomize factor; j's parking query
M	Length of the array in the hashmap
Num	Number of suppliers (drivers)
$Snum$	Number of a supplier's private parking spot
$Dnum$	Number of a driver's query times
Dur	Number of duration of a parking time (hourly)
$Times$	Number of a driver's queries
$hash$	The hash function used in hashmap
Ecc	A coordinate of a elliptic curve point
p, q	Big prime number, order of the elliptic curve
p', q'	Two big prime numbers
A, B, C	Three random numbers in anonymous payment

certain amount of parking fee for the extra parking resources. We denote both supplier and driver as "user" in this paper.

- *Driver* cruises around to find a parking spot in a public parking lot or waits for a private parking spot for which he is willing to pay a parking fee.

3.2 Threat Model

The trusted authority is fully trusted and will not be breached by any adversary. We assume that drivers, suppliers, and the server are honest-but-curious, meaning that they will strictly follow the predesigned scheme, but may also try to pry into others' privacy from available information.

- Server is interested in learning users' locations and identities from parking interactions. It also tries to link a spot location to a supplier, links a location to a driver, and links one user's different messages.
- Supplier may try to obtain the identities of drivers who park in her spot. She may also report inaccurate parking spot information to collect an extra parking fee. We note that if the identity and supplying report of a supplier are not protected, then her neighbors will know the status of this spot and the supplier (whether she is home or not) which leads to a serious privacy violation. Similarly, the drivers can find out which drivers around her are querying for a parking spot. We *do not* consider the physical attack from recording users and their behaviors with private cameras which are not obfuscated because anyone with a smartphone can take a picture anytime and anywhere without being detected and it cannot be prevented.
- Driver may try to obtain the identities of suppliers, park in other drivers' matched parking spots, refuse to pay a negotiated parking fee, or damage a parking spot on purpose.
- External adversary can eavesdrop on communication channels to capture the transmitting messages and violate the privacy. Moreover, an external adversary may try to impersonate a supplier with a parking-spot to trick a driver into paying it a parking fee.

3.3 Design Objectives

- User Authentication: A user should be authenticated before sending a supplying report/parking query, such that no adversary can impersonate a legal user.
- User Privacy: Users' identities are protected from the server, other users and external adversaries. Users' locations are protected from the server, other users and external adversaries, except the matched user. Given two supplying reports/parking queries from one user, no one can link them to the same user.
- Data Confidentiality and Integrity: The contents of supplying reports and protected from the server, other users and external adversaries. All accepted messages should be transmitted without being altered.
- Traceability: TA can trace the real identity of a misbehaved user in case a dispute happens.
- Anonymous payment: The driver's real identity is not disclosed when paying coupons to a supplier and the supplier's real identity is not disclosed when verifying the coupons with the help of the server and cashing coupons at the server. In addition, unlinkability must be protected, meaning any user's coupons cannot be linked together.
- Quick result retrieval: After storing enough information about private parking spots from different suppliers and parking queries from drivers, the server should efficiently match parking queries with supplying reports.

4 Preliminaries

4.1 Bilinear Pairing

A bilinear pairing is a map e: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are cyclic groups of prime order q. $\mathbb{G}_1$ is generated by g, $\mathbb{G}_2$ is generated by $\hat{g}$. The pairing e has the following properties [26]: 1) Computability: for all $g \in \mathbb{G}_1$, $\hat{g} \in \mathbb{G}_2$, $e(g, \hat{g})$ can be computed efficiently; 2) Bilinearity: for all $g \in \mathbb{G}_1$, $\hat{g} \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p^*$, $e(g^a, \hat{g}^b) = e(g, \hat{g})^{a \cdot b}$; 3) Non-degeneracy: for $g \in \mathbb{G}_1$ and $\hat{g} \in \mathbb{G}_2$, $e(g, \hat{g}) \neq 1$.

4.2 Short Randomizable Signature

Short randomizable signature [19] is a new efficient signature scheme which utilizes bilinear pairing and digital signature and it can be converted into a sequential aggregate signature scheme and a group signature scheme. The signature can be randomized and be used as building blocks for many cryptographic primitives to achieve conditional privacy. The detailed functions are as follows:

- **Setup**(k): Given the security parameter k, *Setup* outputs public parameters $p.p. \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$. These bilinear groups must be of type 3. In the following descriptions, we denote $\mathbb{G}_1^* = \mathbb{G}_1 \setminus \{1_{\mathbb{G}_1}\}$.
- **Keygen**$(p.p)$: It selects $\tilde{g} \xleftarrow{\$} \mathbb{G}_2$ and $(x, y) \xleftarrow{\$} \mathbb{Z}_p^2$, computes $(\tilde{X}, \tilde{Y}) \leftarrow (\tilde{g}^x, \tilde{g}^y)$ and sets sk as (x, y) and pk as $(\tilde{g}, \tilde{X}, \tilde{Y})$.
- **Sign**(sk, m): It selects a random number $r \xleftarrow{\$} \mathbb{G}_1^*$ and outputs signature $\sigma \leftarrow (r, r^{x+y\cdot m})$ on a message m.
- *Verify*(pk, m, σ): It parses σ as (σ_1, σ_2) and checks whether $\sigma_1 \neq 1_{\mathbb{G}_1}$ and $e(\sigma_1, \tilde{X} \cdot \tilde{Y}^m) = e(\sigma_2, \tilde{g})$ are both satisfied. If positive, it outputs 1, and 0 otherwise.
- **GSetup**(1^k): The group manager runs the *Setup* and the *Keygen* algorithms to get $sk = (x, y)$ and $pk = (\tilde{g}, \tilde{X}, \tilde{Y})$. It then sets the group public key gpk as pk along with generator $g_1 \in G_1$, and sets his secret key $gmsk$ as sk.
- *PKIJoin*$(i, 1^k)$: The user i generates $(sk_i, pk_i) \leftarrow$ **Keygen**$(p.p)$ and sends pk_i to certificate authority.
- **GJoin**: To join the group, user i first generates a secret $s_i \xleftarrow{\$} \mathbb{Z}_p$ and sends $(\tau, \tilde{\tau}) \leftarrow (g^{s_i}, \tilde{Y}^{s_i})$ along with a signature $\eta \leftarrow$ **Sign**(sk_i, τ) to the group manager. Group manager checks the validity of η and the one of the pair $(\tau, \tilde{\tau})$ by testing whether $e(\tau, \tilde{Y}) \overset{?}{=} e(g, \tilde{\tau})$. Then, user i starts a Schnorr's interactive proof of knowledge [27] of s_i. If everything is checked, group manager generates a random $u \xleftarrow{\$}$ and computes $\hat{\sigma} \leftarrow (\hat{\sigma}_1, \hat{\sigma}_2) \leftarrow (g^u, (g^x \cdot \tau^y)^u)$ which is a valid signature on s_i. Finally, group manager stores $(i, \tau, \eta, \tilde{\tau})$ and sends σ to the user who sets gsk_i as $(s_i, \hat{\sigma}, e(\hat{\sigma}_1, \tilde{Y}))$.
- **GSign**(gsk_i, m): To sign a message m the user first randomizes σ by generating a random t and computing $(\hat{\sigma}_1', \hat{\sigma}_2') \leftarrow (\hat{\sigma}_1^t, \hat{\sigma}_2^t)$ and then computes a signature of knowledge of s_i. To do so, he selects a random $k \xleftarrow{\$} \mathbb{Z}_p$ and computes $c \leftarrow H(\hat{\sigma}_1', \hat{\sigma}_2', e(\hat{\sigma}_1, \tilde{Y})^{k\cdot t}), m)$ for a cryptographic hash function H. Finally, he computes $s \leftarrow k + c \cdot s_i$ and outputs $(\hat{\sigma}_1', \hat{\sigma}_2', c, s) \in \mathbb{G}_1^2 \times \mathbb{Z}_p^2$ as the group signature μ on m.
- **GVerify**(gpk_i, m, μ): To verify a signature $\mu = (\hat{\sigma}_1, \hat{\sigma}_2, c, s)$ on message m, the verifier computes $T \leftarrow e(\hat{\sigma}_1, \tilde{X})^c \cdot e(\hat{\sigma}_2, \tilde{g})^{-c} \cdot e(\hat{\sigma}_1, \tilde{Y})^s$ and then checks whether $c \overset{?}{=} H(\hat{\sigma}_1, \hat{\sigma}_2, T, m)$. If it is valid, it outputs 1, and 0 otherwise.
- **GOpen**$(gmsk, m, \mu)$: To open a signature μ, group manager tests, for all entries $(i, \tau_i, \eta_i, \tilde{\tau}_i)$, whether $e(\hat{\sigma}_2, \tilde{g}) \cdot e(\hat{\sigma}_1, \tilde{X})^{-1} = e(\hat{\sigma}_1, \tilde{\tau}_i)$ holds until it gets a match. It then outputs the corresponding (i, τ_i, η_i) along with a proof of knowledge of a valid $\tilde{\tau}_i$.

5 Anonymous Smart-Parking and Payment Scheme

5.1 Overview of ASAP

During the system initialization phase, suppliers and drivers register with TA. When the driver wants to park his car, he sends his query Que along with a group signature to the server and waits for a result. The server publishes the number of parking queries in different areas, a supplier can choose to provide her private parking spot and send a supplying report Rep along with a group signature to a server. After the server verifies the validity of all reports and queries, it performs matching in the database for each parking query and returns the result to corresponding driver. Finally, when a supplier and a driver reach an agreement, the driver can park in a specific location and they will keep a copy of each other's randomized group signature. Otherwise, the driver informs the server of a negotiation failure and queries for another spot while the supplier's spot is stored in server's database.

5.2 System Initialization

The TA chooses a security parameter k, runs *Setup* to output public parameters $(g, \hat{g}, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, H)$. TA randomly generates $(x, y) \in \mathbb{Z}_p^2$ as group secret keys and computes $(\tilde{X}, \tilde{Y}) \leftarrow (\tilde{g}^x, \tilde{g}^y)$ to set a group public key as $(\tilde{g}, \tilde{X}, \tilde{Y})$.

Each supplier has a unique identifier ID_s, which can be the address of her parking spot. She first randomly picks $s_0 \in \mathbb{Z}_p$ as her secret key and computes $D_s = g^{s_0}$ as her public key. Then she randomly chooses a secret $s_1 \in \mathbb{Z}_p$ and sends pair $(\tau, \tilde{\tau}) \leftarrow (g^{s_1}, \tilde{Y}^{s_1})$ with signature $\eta \leftarrow$ **Sign**(s_0, τ) to TA. After successfully checking the validity of η, testing whether $e(\tau, \tilde{Y}) \overset{?}{=} e(g, \tilde{\tau})$ and finishing an interactive proof of knowledge of s_1, TA will randomly generate a random $u \in \mathbb{Z}_p$ to calculate

$$(S_1, S_2, S_3) \leftarrow (g^u, (g^x \cdot \tau^y)^u, e(S_1, \tilde{Y})). \tag{1}$$

Finally, TA stores $(ID_s, g_{s_1}, \eta, \tilde{\tau})$ in the database and sends (S_1, S_2, S_3) to supplier ID_s. Supplier ID_s sets her group secret key

$$gsk_{ID_s} = (s_1, S_1, S_2, S_3), \tag{2}$$

and the group public key

$$gpk_{ID_s} = g^{s_1}. \tag{3}$$

Similar for each driver who has a unique identifier ID_d, which can be the license plate, he will have his secret key d_0, public key $D_d = g^{d_0}$, group secret key $gsk_{ID_d} = (d_1, D_1, D_2, D_3)$ and group public key $gpk_{ID_d} = g^{d_1}$.

5.3 Driver Querying

When driver j with ID_j is driving on a road and looking for a parking spot, he sends a parking query to the server. Specifically, the driver utilizes his smartphone to generate the basic parking information, including pseudo-id Pid_j, desired parking location Pa_j, querying time tt_1, expired time tt_2. He performs the following steps to form a parking query:

- Randomly choose $k_j \in \mathbb{Z}_p$ to generate a temporary public key $K_j = \tilde{g}^{k_j}$ which is used for the communication between herself and the supplier to find.
- Encrypt (Pa_j, K_j) by randomly choosing $r_0 \in \mathbb{Z}_p$ and computing

$$C_{j1} = \tilde{g}^{r_0}, \tag{4}$$

$$C_{j2} = (Pa_j || K_j) \cdot \tilde{Y}^{r_0}. \tag{5}$$

- Randomize (D_{j1}, D_{j2}, D_{j3}) by selecting $(r_3, r_4) \in \mathbb{Z}_p^2$ to calculate

$$(D'_{j1}, D'_{j2}, D'_{j3}) \leftarrow (D_{j1}^{r_3}, D_{j2}^{r_3}, D_{j3}^{r_3 r_4}), \tag{6}$$

$$c_j = H(D'_{j1}, D'_{j2}, D'_{j3}, Pid_j, Pa_j, tt_1, tt_2, K_j), \tag{7}$$

$$ss_j = r_4 + c_j d_1, \tag{8}$$

where d_1 is the secret that driver j chose in the system initialization phase and $(D'_{j1}, D'_{j2}, c_j, ss_j)$ is a valid signature on $(Pid_j, Pa_j, tt_1, tt_2, K_j)$.

Then, driver j stores (k_j, K_j) and sends the parking query Que_j to the server

$$Que_j = (Pid_j, tt_1, tt_2, C_{j1}, C_{j2}, D'_{j1}, D'_{j2}, c_j, ss_j). \tag{9}$$

Upon receiving the parking query Que_j, the server first decrypts C_{j1} and C_{j2} to obtain $Pa_j||K_j = C_{j2}C_{j1}^{-y}$ and verifies the validity of the signature $(D'_{j1}, D'_{j2}, c_j, ss_j)$ by computing

$$D'_j = e(D'_{j1}, \tilde{X})^{c_j} \cdot e(D'_{j2}, \tilde{g})^{-c_j} \cdot e(D'_{j1}, \tilde{Y})^{ss_j}, \tag{10}$$

and checks

$$c_j \overset{?}{=} H(D'_{j1}, D'_{j2}, D'_j, Pid_j, Pa_j, tt_1, tt_2, K_j). \tag{11}$$

If it is not valid then the server returns failure and aborts; otherwise, the server looks up the database to find a parking spot nearby. If there is no match, the server disseminates the parking task (e.g., "Parking spot needed!!!") to the suppliers in area Pa_j and these suppliers can answer to the driver by supplying a vacant parking spot. Here, we note that the public list, including each driver's cloaked location and tasking time, is not recommended because there is a potential privacy leakage in publishing such list [18].

5.4 Supplier Reporting

If supplier i with ID_i is willing to offer her private parking spot, she sends a supplying report to the server after anonymous authentication similar to what driver performs in previous section. Specifically, the supplier utilizes her smartphone to generate the basic supplying information, including pseduo-id Pid_i, cloaked parking location $Supp_i$, supplying time t_1, expired time t_2, spot price pr_i, and performs the following steps to form a supplying report:

- Randomly choose $k_i \in \mathbb{Z}_p$ to generate a temporary public key $K_i = \tilde{g}^{k_i}$ which is used for the communication between himself and the driver to come.
- Encrypt $(Supp_i, K_i, pr_i)$ by randomly choosing $r_0 \in \mathbb{Z}_p$ and computing ciphertext $C_{i1} = \tilde{g}^{r_0}$, $C_{i2} = (Supp_i||K_i||pr_i) \cdot \tilde{Y}^{r_0}$.
- Randomize (S_{i1}, S_{i2}, S_{i3}) by selecting $(r_1, r_2) \in \mathbb{Z}_p^2$ to calculate $(S'_{i1}, S'_{i2}, S'_{i3}) \leftarrow (S_{i1}^{r_1}, S_{i2}^{r_1}, S_{i3}^{r_1 r_2})$, $c_i = H(S'_{i1}, S'_{i2}, S'_{i3}, Pid_i, Supp_i, t_1, t_2, K_i, pr_i)$, $ss_i = r_2 + c_i s_1$, where s_1 is the secret that supplier i chose in the system initialization phase and $(S'_{i1}, S'_{i2}, c_i, ss_i)$ is a valid signature on $(PID, Supp_i, t_1, t_2, K_i, pr_i)$.

We note that the cloaked location is an experience-based parameter used for privacy protection and published by the server. Then, supplier i stores (k_i, K_i) and sends her supplying report Rep_i to the server

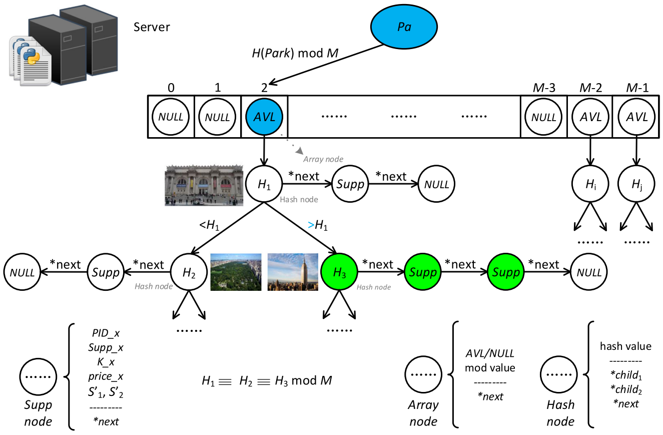


Fig. 2. Matching query with supply in the Hashmap.

$$Rep_i = (Pid_i, t_1, t_2, C_{i1}, C_{i2}, S'_{i1}, S'_{i2}, c_i, ss_i). \tag{12}$$

Upon receiving the supplying report Rep_j, the server first decrypts C_{i1} and C_{i2} to obtain $K_i||Supp_i||pr_i = C_{i2}C_{i1}^{-y}$ and verifies the validity of the signature $(S'_{i1}, S'_{i2}, c_i, ss_i)$ by computing

$$S'_i = e(S'_{i1}, \tilde{X})^{c_i} \cdot e(S'_{i2}, \tilde{g})^{-c_i} \cdot e(S'_{i1}, \tilde{Y})^{ss_i}, \tag{13}$$

and checks whether

$$c_i = H(S'_{i1}, S'_{i2}, S'_i, Pid_i, Supp_i, t_1, t_2, K_i, pr_i). \tag{14}$$

If it is not valid then the server returns failure and aborts; otherwise, it stores $(Pid_i, Supp_i, K_i, pr_i, S'_{i1}, S'_{i2})$ in the database. Specifically, we assume that the server can collect parking spots in a city. The whole city area is first partitioned according to M neighboring square cells and published online since this information is not sensitive. A hashmap consists of an array of M *Array node*: each location in the array is a tree root; every spot information is first hashed according to the grid identity and stored into a node in the tree; if two nodes has same hash value, it means that they are within a same grid (but the accurate locations are hidden); the spot information is hashed to be stored along the tree for the second time according to specific requirements, such as price. For instance, *Array nodes* with "NULL" identification are pre-stored in each location of the array. The first parking spot location $Supp_j$ is hashed into h_i mod M and then $(Pid_i, Supp_i, K_i, pr_i, S'_{i1}, S'_{i2})$ is stored in a new *Supp node*. This new node is added to the tree under one specific *Arrary node* and *Arrray node*'s identification is changed into "AVL". Subsequent parking spot information with the same hashed value is stored and inserted to the linked list. The hashmap is shown in Fig. 2 and the matching details are discussed in next section.

5.5 Result Retrieving

After storing enough reports about parking spots, the server can retrieve a corresponding parking spot for a querying driver. The server hashes and mods a parking location Pa into M and checks whether the corresponding node in the array is AVL. If not, the server returns "No available spots."; otherwise, it continues to search along the tree downwards. Hash value of $Park'$ is archived in each *Hash node* which has two child nodes and one brother node. The hash value in left/right child node is smaller/bigger than that of parent node. The brother node stores the location which is in the same area indicated by the $Hash\ node$. When the server finds a suitable result $Res = (S_1, S_2, PID, Supp, K, price)$ (if there are a set of results, it will sort them by the price and return a certain number of results for selection, such as 5), it returns it to driver j through encrypting Res with K_j.

If driver j accepts the parking result, he will first communicate with supplier i through the server with Pid_i and encrypt his messages with K_i. The supplier i will encrypt his specific location with K_j and send it to j. We note that j actually parks in the this spot, no identity-related information (e.g., name, cellphone number, color, type of clothes) of i or j needs to be revealed out of privacy concerns and we discuss it in Section 5.4.

If i agrees to provide the parking spot, she will send a confirmation message (e.g., Pid_j, Pa_j) to the server which will send j's $(D'_{j1}, D'_{j2}, Pid_j, K_j)$ to i and delete the corresponding $Supp\ node$ in the database. If the supplier is not able to respond to the parking query, or not willing to provide her spot for some reasons (e.g., the spot will be used by himself or her friends, the spot needs maintenance), the server awaits a response from the supplier for a time period T and search another parking spot for the driver if no response is received after T expires. The cloaking technique is applied as follows:

- First, we use the city map of Beijing [28] and divide it into M neighboring cells which are published online since they are not sensitive. Second, we expand a user's specific location into a grid identity to protect the location privacy. The driver and supplier will send a grid identity to the server for matching a supplier. Then, the server will return a supplying report to the driver. Third, only when a negotiation consensus is reached, will the supplier and the driver know each other's specific location.
- A hashmap which consists of an array of M $Arraynode$s: each location in the array is a tree root; every spot information is hashed according to the grid identity and stored into a node in the tree; if two nodes have same hash value, it means that they are within a same cell; the spot information is hashed into the tree for the second time according to the specific requirements, such as price. When a parking query including Pa is received, the server can find a matching spot in the hashmap by hashing Pa and searching along a tree under a matching array node. We note that only when the variables are the same, will the hash value be the same. The cloaking technique helps two variables with close locations be hashed into a same output.

5.6 Identity Disclosing

While providing private parking spots to unacquainted drivers, people who share the same living or working community with the supplier have concern about the inconvenience or threat from the outsider. On the other hand, a misbehaved supplier may provide inaccurate or others' parking spots to gain profits. These situations will happen when there is a dispute. Therefore, TA should be able to track a targeted user.

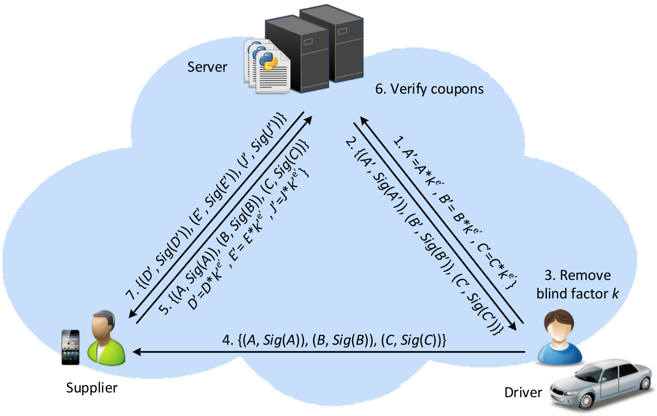

Server
6. Verify coupons
1. A'=A*k^{e'}, B'=B*k^{e'}, C'=C*k^{e'}}
2. {(A', Sig(A')), (B', Sig(B')), (C', Sig(C'))}
3. Remove blind factor k
4. {(A, Sig(A)), (B, Sig(B)), (C, Sig(C))}
5. {(A, Sig(A)), (B, Sig(B)), (C, Sig(C))}
D'=D*K'^{e'}, E'=E*K'^{e'}, J'=J*K'^{e'}}
7. {(D', Sig(D')), (E', Sig(E')), (J', Sig(J'))}
Supplier
Driver

Fig. 3. Anonymous payment.

Utilizing group signature, TA can trace the real identity of a driver using his signature (D'_1, D'_2). For all entries $(i, \tau_i, \eta_i, \tilde{\tau}_i)$, it tests whether

$$e(D'_2, \tilde{g}) \cdot e(D'_1, \tilde{X})^{-1} = e(D'_1, \tilde{\tau}_i), \qquad (15)$$

until a match is found. It then inserts the matched $(i, \tau_i, \eta_i, \tilde{\tau}_i)$ in its blacklist, sets a period of dormancy (e.g., 10 minutes, 30 minutes, and one hour) for this driver, and sends $\tilde{\tau}_i$ to the server. The server will immediately detects this driver by checking Equation (20) if the driver submits a parking query again within the timeout. After the dormancy ends, the driver has to re-register to TA and obtains new keys and signatures. Similar for a supplier, TA has the identity revealing ability.

5.7 Anonymous Payment

After the driver leaves the parking spot, he should pay a parking fee to the supplier with the help of the server in an anonymous way. The server, driver and supplier perform the following steps to co-complete an anonymous payment. The general idea of anonymous payment is shown in Fig. 3.

- The server chooses two prime numbers p', q', computes $n' = p' * q'$ and gets $\mathbb{G}$. A public key e' is randomly chosen that is prime to $(p' - 1) * (q' - 1)$ and the server computes secret key d satisfying $e' * d' = 1(\mathrm{mod}(p' - 1) * (q^{'} - 1))$. We assume that there are there values($1, $2, $5) of coupon and the server repeats the procedure to get different set of keys (CK_1, CK_2, CK_3). The server signs a coupon with different CK according to value.
- The driver sends $(ID, V, Set(V), H(ID, V, Set(V)))$ to the server and pays him what is equivalent to V which is the total values of coupons he needs to buy. For example, $V = 12$ and driver needs at least three coupons (5+5+2). First, he chooses three random number (A, B, C) with a blind factor F, and computes: $Set(V) = \{A', B', C'\}$, $A' = A * F^{e'}$, $B' = B * F^{e'}$, $C' = C * F^{e'}$, and sends them to the server while indicating the corresponding coupon value of (A', B', C').
- The server verifies the hashed value of $(ID, V, Set(V))$ and chooses signing key to sign

$Set(V)$ according to V if it is valid. The server computes $Sig(H(A')) = (H(A) * F^{e'})^{d'} = H(A)^{d'} \bmod n$ with similar $Sig(H(B'))$, $Sig(H(C'))$). Then server attaches current time ct and a coupon id c_{id} to $\{(Sig(H(A')), Sig(H(B')), Sig(H(C'))\}$ which are stored in the database. The coupons are stored in different sub-databases according to ct for better management and quick verification in next step.

- The driver verifies whether the signature is valid by verifying $(Sig(H(A')))^e \overset{?}{=} (H(A) * F^e)$. If the verification holds, driver computes $Sig(H(A)) = Sig(H(A')) * F^{-1}$ and stores $\{(A, Sig(H(A))), (B, Sig(H(B))), (C, Sig(H(C)))\}$ as coupons. When he is leaving the spot, he sends encrypted coupons to the supplier.
- The supplier decrypts the encrypted coupons and asks the server to check whether these coupons are valid. The server searches database to see whether they have been used before. Each coupon has a unique identification and is stored according to the timestamp ct such that the server can quickly locate it and check whether it has been used. If they are not valid, the server will give a notice to the supplier to recharge the supplier; otherwise, the server stores the coupons in the coupon database and marks it as used. Then the server informs the supplier that the coupons are valid and asks him to generate equivalent coupons to be blind-signed. Finally, the supplier performs the similar steps as driver buys coupon in the first place.
- The supplier can cash them from the server after collecting enough coupons to obfuscate parking time points.

Note on Payment Time. Different cities have different time points of parking fee payment, and they can also vary even in the same city. In Singapore [29] and Ottawa [30], the parking fee is charged before the driver parks his vehicle, while in Beijing [31] and Shanghai [32], it is charged after the driver finishes parking. Therefore, it is really a matter of convention or regulation regarding when to pay the parking fee. ASAP can support these two scenarios at the same time. If we choose to execute the payment phase at the end of parking: the driver has to send the coupons when he is leaving the spot. If we choose to execute the payment phase at the beginning of parking: the driver has to pay the parking fee beforehand after the supplier and driver reach a negotiation consensus.

6 Security and Privacy Analysis

6.1 User Authentication

Definition 1 (LRSW Assumption 1). *Let $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ be a bilinear group setting of type 3, with $g, \tilde{g}$ be a generator of $\mathbb{G}_1, \mathbb{G}_2$ respectively. For $x, y \in \mathbb{Z}_p, X = g^x, Y = g_y, \tilde{X} = \tilde{g}^x, \tilde{Y} = \tilde{g}^y$, we define the oracle $\mathcal{O}(m)$ on input $m \in \mathbb{Z}_{\mathbb{p}}$ that chooses a random $h \in \mathbb{G}_1$ and outputs a signature $Sig = (h, h^{x+my})$. Given $(g, Y, \tilde{g}, \tilde{X}, \tilde{Y})$ and unlmited access to $\mathcal{O}$, no adversary can efficiently generate such a pair $(h \neq 1_{\mathbb{G}_1})$ for a new $m*$ which has not been asked to $\mathcal{O}$. [19]*

Theorem 1. *The existential unforgeability under chosen message attacks (EUF-CMA) of the supplier/driver's blind signature (S_1, S_2) holds under the LRSW Assumption 1.*

Proof. Let Π' be the signature scheme in LRSW Assumption 1, and Π be our proposed ASAP. Let $\mathcal{A}$ be a probabilistic polynomial-time adversary attacking Π' with $q = q(k)$ an upper bound on the number of queries that A makes to a supplier $\mathcal{O}$. We make three simplifying assumptions without loss of generality here. First, A makes any given query to $\mathcal{O}$ only once. Second, A never queries m after being given a signature $sig = (sig_1, sig_2)$ on m with verification $\mathcal{V}(m, sig_1, sig_2) = 1$ where $\mathcal{V}$ is $e(Sig_1, \tilde{X} \cdot \tilde{Y}) \overset{?}{=} e(Sig_2, \tilde{g})$. Third, A outputs a forged signature (sig_1', sig_2') on message m' if $\mathcal{V}(m', sig_1', sig_2') = 1$. Now we construct an efficient adversary $\mathcal{A}'$ that runs $\mathcal{A}$ and attacks Π'.

Algorithm $\mathcal{A}'$. The algorithm is given pk in Π which contains the public parameters $p.p. \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ and public key $pk = (\tilde{g}, \tilde{X}, \tilde{Y})$ and access to an oracle $\mathcal{O}$.

1. Choose uniform $i \in \{1, 2, \ldots, q\}$.
2. Run $\mathcal{A}(p.p, pk)$. When $\mathcal{A}$ makes jth random-oracle query m_j, answer it as follows: If $j = i$, query $\mathcal{O}$ to obtain a transcript (m, sig_1, sig_2) of an honest execution of the scheme and return the signature (sig_1, sig_2). If $j \neq i$, choose uniform $sig_1, sig_2 \in \mathcal{G}_1$ and return them.
3. If $\mathcal{A}$ outputs a forged signature (sig_1', sig_2') on a message m', check whether $\mathcal{V}(m', sig_1', sig_2') = 1$ and $m' \neq m_j$. If so, output (m', sig_1', sig_2'). Otherwise, abort.

The view of $\mathcal{A}$ when run as a subroutine by $\mathcal{A}'$ in experiment $\textsf{Sig-forge}_{\mathcal{A}',\Pi'}$ is identical to the view of $\mathcal{A}$ in experiment $\textsf{Sig-forge}_{\mathcal{A},\Pi}$. If $\mathcal{V}(m', sig_1', sig_2') = 1$ and m' has never been asked to the signing oracle, this is a valid forgery. Therefore, we have $\Pr[\textsf{Sig-forge}_{\mathcal{A}',\Pi'}] \geq \frac{1}{q(k)} \cdot \Pr[\textsf{Sig-forge}_{\mathcal{A},\Pi}]$. If Π' is secure, then $\Pr[\textsf{Sig-forge}_{\mathcal{A}',\Pi'}$ is negligible. Since $q(k)$ is polynomial, it implies that $\Pr[\textsf{Sig-forge}_{\mathcal{A},\Pi}]$ is also negeligible. We complete the proof. □

6.2 User Privacy

First, a user uses an anonymous credential to prove a qualification in the system such that the server cannot know the real identity of the user. Second, the user selects a random number r_1 to randomize the credential in each report/query such that any two of them will not be linked together. Therefore, the identity privacy is guaranteed. We aim to protect the overall user privacy: users' locations are protected from the server, other users and external adversaries, except the matched user. For instance, a driver will know the location of a matched supplier and vice versa, while the server, other users and external adversaries cannot know this location. However, even if the driver knows the physical location of the spot, the anonymity of the supplier is still guaranteed because the spot usually does not have identical information linked to its owner, and the driver needs more background information to know the supplier's identity. The supplier cannot know this driver's parking locations every time. Even in probabilistic encryption, two different plaintexts under a same encryption key have a small possibility of sharing a same ciphertexts. Moreover, the driver has to know where the spot is on order to park his vehicle, which is a basic requirement in a system such as private parking and carpooling, and the system cannot function well if this requirement is not met.

6.3 Data Confidentiality and Integrity

For the supplying reports and parking queries, we adopt the Elgamal encryption [33] scheme to encrypt them. Specifically, $(Supp||K||pr)$ and $(Pa||K)$ are protected by a temporary public key r_0 and the server's secret key. Since the Elgamal encryption is proved to be secure, the supplying reports and parking queries are well-protected against the server, other users and external adversaries. The messages between the server and users are authenticated using the Schnorr signature scheme which is secure under the discrete logarithm assumption, thereby, the integrity of the messages is guaranteed.

6.4 Traceability

The TA can trace the real identities of the querying driver who sends a parking query or the supplier who offers a parking spot in case a dispute happens. Using received signatures (Sig_1, Sig_2) and all entries $(i, \tau_i, \eta_i, \tilde{\tau}_i)$, TA can test $e(Sig_2, \tilde{g}) \cdot e(Sig_1, \tilde{X}) \overset{?}{=} e(Sig_1, \tilde{\tau})$ holds to disclose the real identity of the driver or the supplier. Moreover, only the TA can recover the supplier/driver's identity from their signatures because $\tilde{\tau}$ is only known to the TA besides the identity owner.

6.5 Anonymous payment

(1) When a driver is leaving a spot, he sends to supplier encrypted coupons. Since the driver utilizes blind signatures to obtain valid coupons, the server cannot identify him when these very coupons are being verified such that the driver anonymity here is protected. Since the coupons are encrypted by the supplier's public key, other suppliers, drivers and external adversaries cannot obtain the plaintext. The driver unlinkability is protected by the randomized coupons.

(2) When the supplier is verifying the coupons, she sends the encrypted coupons to the server which blind-signs new coupons if the received coupons are valid. Since the coupons are encrypted by the server's public key, other suppliers, drivers and external adversaries cannot obtain the plaintext coupons. The supplier unlinkability is protected due to the randomized coupons.

(3) When the supplier is cashing coupons at the server, she has obtained new blind-signed coupons from the server and she can wait to collect enough coupons to obfuscate the original parking time. Therefore, her real identity and supplier unlinkability are protected.

7 Performance Analysis

Our prototype of the server in ASAP runs on a desktop with two Intel-Xeon-E5620 processors, a 12 GB memory and Microsoft Windows 7 professional operating system. The parking locations are mapped to a city map of Beijing and the dataset we reference to is the Microsoft's project T-Drive [34] as we did in our previous work [35]. First, we manually count the number of private parking spots in twenty residential communities around our workplace. For instance, there are three communities (north to south) between 3rd Ring Road and South College Road, and (from west to east) between South Zhongguancun Street and Zaojunmiao Road. The number of private spots in each community are set to 40, 10 and 30. Second, some residential communities have their own LED boards showing the total and remaining number of private parking spots. Third, in the future, we can acquire this number based on the number of suppliers' reports. We also use the Microsoft's project T-Drive dataset to obtain the trajectories of taxies to acquire the approximate density of vehicles. For example, a higher trajectory number indicates a higher traffic congestion, meaning more parking demands are generated in this area.. The cryptographic toolset we used is MIRACL version 7.0.0 [36]. The length of p is 160-bit and the elliptic curve is defined as $y^2 = x^3 + 1$ over $\mathbb{F}_q$, where the length of q is 512-bit. In each set of experiments with different number of suppliers (drivers) Num, we took an average of 100 runs with the array length $M = 1000$. The supply data (each supplier's number of private spots $Snum$) and query data (each driver's query number in one day $Qnum$ and the duration of a parking Dur (hours)) we used as input to our system are not random, and we simulate them as close to real life scenarios as possible. The detailed parameter values are listed in Table 2.

TABLE 2
The Parameter Settings

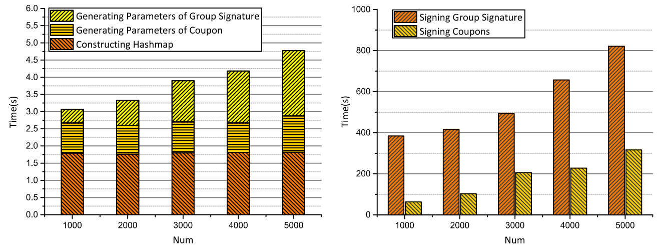
Parameter	Value
M, Num	$1000, [1000, 5000]$
$Snum, Dnum, Dur$	$1, [1, 5], (0, 2]$
$\#Report, \#Query, hash$	$[1, 10]; SHA256$
p, q, Ecc	$p\vert = 160, \vert q\vert = 512, \vert Ecc\vert = 168$
PID, t, tt	$\vert PID\vert = 2^{13}, \vert t\vert = \vert tt\vert = 2^{12}$
ss, p', q'	$\vert ss\vert = 2^8, \vert p'\vert = \vert q'\vert = 160$
A, B, C	$\vert A\vert = \vert B\vert = \vert C\vert = 2^{15}$

7.1 Computational Costs

We now simulate the computational costs for a supplier, a driver, and the server. We compare ASAP with VSPN [12], PRIN [13], and CPARN [14] because they are adjacent works focused on privacy-preserving parking or navigation scheme design in vehicular networks and the main techniques used in the three scheme are cryptographic tools and data structure. Add_2, Mul_2, Exp_1, Exp_2, H, BP, Mul' and Exp' denote the operations of addition in $\mathbb{G}_2$, multiplication in $\mathbb{G}_2$, exponentiation in $\mathbb{G}_1, \mathbb{G}_2$, hash function, bilinear pairing, multiplication and exponentiation in $\mathbb{G}$.

7.1.1 Trusted Authority

We first look at the setup time for the server before it answers any queries. When signing group signatures, TA executes $Exp_1 + 4Exp_2 + Mul_2 + 3BP$ operations to generate keys for each supplier and driver. When signing coupons, TA executes $2Exp' + Mul'$ operations to sign one coupon. Fig. 4 shows the average time for the server to generate parameters of group signature, generate parameters of coupon and construct hashmap. Fig. 4b shows the average time for the server to sign group signatures for users and sign coupons for drivers. From the figures above, we can see that the time for parameter generation and hashmap construction is less than five seconds while the time of sign



(a) Generating Parameters and Constructing Hashmap (b) Signing Group Signatures and Coupons

Fig. 4. Setup time.

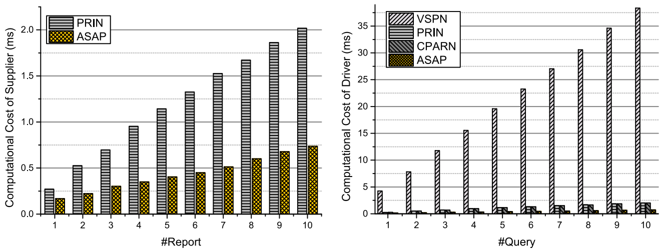
group signatures and coupons, which increases with Num, makes up the majority of setup time.

7.1.2 Supplier

We now look at the computational cost for the supplier in supplier reporting phase and compare with PRIN [13], since only this scheme has similar supplier entity in their system model in related work [12], [13], [14]. Each supplier has to compute $Exp_1 + 4Exp_2 + Exp_3 + 2Mul_2 + Add_2 + H$ to form a supplying report. The operations of a supplier in this phase in PRIN and ASAP are similar, except that encryption is not required in the former. The comparison result of computational cost for the supplier is shown in Fig. 5a, indicating that the total time of a supplier sending her reports in ten round is less than 1 ms and it is lower than that of PRIN. The supplier has to compute three new blinded coupons, which can be done in advance, and computations are same with the driver.

7.1.3 Driver

We now look at the computational cost for the driver in driver querying phase and compare with VSPN [12], PRIN [13], and CPARN [14]. Each driver has to compute $Exp_1 + 4Exp_2 + Exp_3 + 2Mul_2 + Add_2 + H$ to form a parking query. The operations of a driver in this phase in PRIN and CPARN are similar to ASAP, except that an AES encryption is required in the former two schemes. Only one asymmetric encryption is required for the driver in VSPN. The comparison result of computational cost for the driver is shown in Fig. 5b, indicating that ASAP achieve the minimum computational cost among the four comparison, and the total time of a driver sending his queries in ten round is less than 1 ms.



(a) Supplier (b) Driver

Fig. 5. Comparison of computational costs.

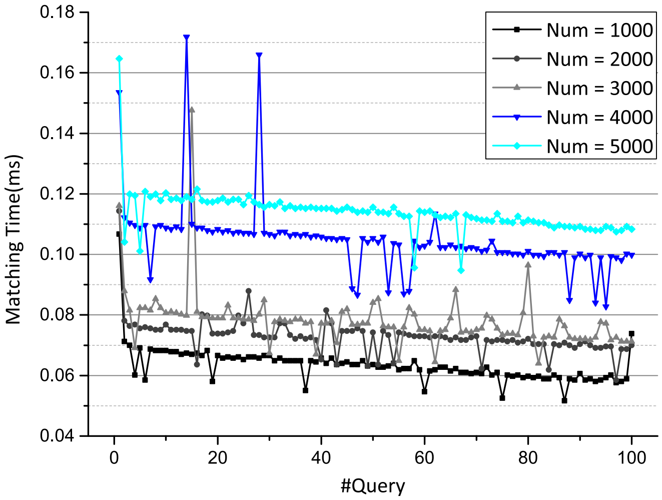


Fig. 6. Matching time.

7.2 Result Retrieving

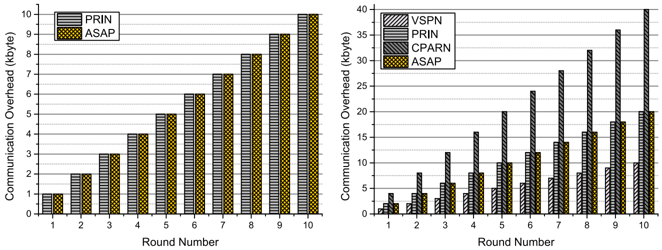
After the setup is finished, we see how quickly the server responses to a parking query. Now the server has collected enough reports and constructed a hashmap storing parking locations $Supp$s. We conducted the matching experiments under five settings where $Num = 1000, 2000, 3000, 4000, 5000$ respectively, and the number of queries is 100. As depicted in Fig. 6, the overall matching time is less than two milliseconds. Since we used binary tree in the hashmap, the average (worst) matching time is $O(log_2 Num)(O(Num))$ which is the same for deleting. We insert new $Supp\ node$ right after $Hash\ node$ such that the average inserting time is $O(log_2 Num)$. Some outliers with higher time value means the corresponding $Supp$ is stored in the lower level in the tree.

7.3 Communication Overhead

For a supplying report: it contains a pseudo-id PID, two time stamps t, two ciphertext C_{i1}, C_{i2}, and a signature $(S'_{i1}, S'_{i2}, c_i, ss_i)$. The communication overhead is $2^{13} + 2 * 2^{12} + 2 * 168 + 2 * 168 + 2 * 2^8 \approx 2^{14}$ bits. For a parking query: it contains a pseudo-id PID, two time stamps t, two ciphertext C_{j1}, C_{j2}, and a signature $(S'_{j1}, S'_{j2}, c_j, ss_j)$. The communication overhead is $2^{13} + 2 * 2^{12} + 2 * 168 + 2 * 168 + 2 * 2^8 \approx 2^{14}$ bits. Note that VSPN and CPARN do not have the supplier in their models, thus we do not compare with these two schemes regarding the communication overhead of this entity. In VSPN, the driver requests TA for a navigation session number, sends a real identity RRID and a ciphertext to RSU. In PRIN, the supplier sends a sequence number, two timestamps, a traffic information report, and s signature to the RSU; the driver sends a sequence number, two timestamps, three ciphertext, and a signature to the RSU. In CPARN, the driver sends a random number, a current time, an anonymous credential, a zero-knowledge proof, three ciphertext, and a signature to the RSU. The comparison result is shown in Figs. 7a and 7b, indicating can see that the communication overhead of ASAP is comparatively moderate among the four schemes.

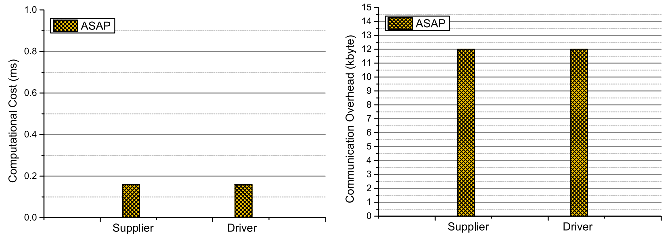
7.4 Performance in Anonymous Payment Phase

For instance, the parking fee is \$8, and the driver sends his coupon $\{(rand_1, Sig(H(rand_1))), (rand_2, Sig(H(rand_2))), (rand_3, Sig(H(rand_3)))\}$ to the supplier, which will send the coupons to the server. After the server checks the validity of the coupon, it returns three new blind-signed coupons to the supplier for future cashing. The length of big prime number p' and q' is 160 bits. From results in Fig. 8, we can



(a) Supplier (b) Driver

Fig. 7. Comparison of communication overhead.



(a) Computational Cost (b) Communication Overhead

Fig. 8. Computational cost and communication overhead in anonymous payment phase.

see that a supplier and a driver have the similar computational cost and communication overhead which are 0.16 ms and 12 kbytes.

8 Discussion

Supplier Motivation. In a typical data collecting and sharing platform, the workers or reporters have incentives to participate because they will get rewards from the system manager for valid contributions. In ASAP, suppliers earn parking fees which is the incentive by providing private parking spots. Therefore, in order to recruit more suppliers, the system can provide extra benefits (e.g., gas coupons) to suppliers and the drivers can behave according to the negotiation result such that the system will enter a healthy circulation and assist society in trust establishment. In addition, if more than one spots are available for one driver, the suppliers can compete to be selected by the driver according to a bidding system [37].

Combining Techniques. It is difficult to combine the techniques in ASAP. Imagine that when the system attempts to achieve payment process between suppliers and drivers, an intuitive method is that drivers pay by e-cash. A digital signature can be added to protect the integrity of the e-cash. However, when a driver buys an e-cash with his credit card or debit card, his real identity is exposed at the same time. The privacy cannot be preserved because the system can record the card number and the e-cash ID in its database. Another alternative method is to use anonymous payment technology. However, if we use anonymous payment such as Bitcoin[25] and TOR [38], there will be disadvantages: for the Bitcoin approach, the suppliers and drivers have to register a Bitcoin account in the first place; the driver must wait for at least 1 hour (6 blocks) to be found in order to gain high confidence that the transaction with a driver is actually acknowledged in the network [39]. Therefore, it is not practicable for the supplier to actually receive the payment. For the TOR approach, it will raise more communication overhead since layers of encryption and decryption are required in transmission.

Securing Parking Spots. One supplier's renting out neighbor's parking spots be prevented by using a remote control ground parking lock [40]. Only after the supplier agrees to lend her paring spot to a driver, she will unlock the parking lock for the driver, or tells the driver how to unlock it. Furthermore, this behavior will incur legal sanction because private parking spots are personal property as well.

Other Considerations. If a driver does not pay the parking fee, the supplier can file a complaint to the TA, if the driver cannot provide the payment proof received from the supplier, the TA deems the driver as a violator, recovers his real identity, and inserts the driver into a blacklist. The parking duration time can be calculated by a camera at the private area entrance/exit gate. There are two types of cameras: ones installed by the government and ones installed by residents. For the first type, we trust the administrative personnel which has professional ethics and their regular monitoring through cameras is not considered as a privacy violation to the residents (ASAP users) whether the cameras are obfuscated. Even if someone misuses the camera and violates users' privacy, he/she will be punished by the law. Meanwhile, some cameras only aim at detecting certain users, such as the roadside cameras only take pictures of overspeeding vehicles. For the second type, the cameras which are obfuscated can be used in ASAP, since the face and plate number will not be clearly recorded; we do not consider the physical attack from cameras which are not obfuscated because anyone with a smartphone can take a picture anytime and anywhere without being detected and it cannot be prevented. Meanwhile, not all private parking spots have a secret camera.

9 Conclusion and Future Work

In this paper, we proposed ASAP to enable the cruising driver to find a parking spot and supplier to make a profit from providing private parking resources. The parking spots are better utilized and traffic congestion is further reduced. A supplier and driver can anonymously send a supplying report and a parking query to the server. Meanwhile, a trusted authority is able to disclose a user's identity if a dispute occurs and users achieve anonymous payment with E-cash. Our scheme also supports finding public parking spot which only needs to add a counting item in the hashmap.

For the future work, first, we will consider detecting location attack from drivers in advance, meaning a driver may send a parking query to the server long he arrives at the destination area, and the system should be able to filter out this query and guarantee system fairness since other drivers in this areas now need parking spots more. Second, we will design a privacy-preserving smart-parking and payment scheme based on fog computing [41] to achieve a more efficient [42], [43] smart-parking and payment scheme for suppliers and drivers. In this way, the suppliers and drivers will be matched locally by fog nodes, and the reports and queries will not be uploaded to a remote server, such that the response time, network bandwidth and public traffic congestion caused by cruising vehicles will be further reduced.

Acknowledgments

This work is partially supported by China National Key Research and Development Program No. 2016YFB0800301.

References

[1] P. White, “No vacancy: Park slopes parking problem and how to fix it.” (2007). [Online]. Available: https://www.transalt.org/news/releases/126

[2] (2017). [Online]. Available: http://www.scmp.com/news/china/society/article/2118751/beijing-looks-smart-solutions-solve-citys-parking-problems

[3] (2016). [Online]. Available: http://www.globaltimes.cn/content/963391.shtml

[4] D. Shoup, “Cruising for parking,” *Transport Policy*, vol. 13, no. 6, pp. 479–486, 2006.

[5] J. Ni, K. Zhang, X. Lin, and X. Shen, “Securing fog computing for internet of things applications: Challenges and solutions,” *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 601–628, Jan.–Mar. 2018.

[6] C. Huang, D. Liu, J. Ni, R. Lu, and S. Shen, “Reliable and privacy-preserving selective data aggregation for fog-based IoT,” in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–6.

[7] M. Mahmoud, K. Rabieh, A. Sherif, E. Oriero, M. Ismail, E. Serpedin, and K. Qaraqe, “Privacy-preserving fine-grained data retrieval schemes for mobile social networks,” *IEEE Trans. Depend. Secure Comput.*, 2017.

[8] R. Lu, X. Lin, H. Zhu, and X. Shen, “SPARK: A new VANET-based smart parking scheme for large parking lots,” in *Proc. IEEE INFOCOM*, 2009, pp. 1413–1421.

[9] R. Lu, X. Lin, H. Zhu, and X. Shen, “An intelligent secure and privacy-preserving parking scheme through vehicular communications,” *IEEE Trans. Veh. Technol.*, vol. 59, no. 6, pp. 2772–2785, Jul. 2010.

[10] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrashekharan, W. Z. Xue, M. Gruteser, and W. Trappe, “ParkNet: Drive-by sensing of road-side parking statistics,” in *Proc. ACM Int. Conf. Mobile Syst. Appl. Services*, 2010, pp. 123–136.

[11] S. Nawaz, C. Efstratiou, and C. Mascolo, “ParkSense: A smartphone based sensing system for on-street parking,” in *Proc. ACM Annu. Int. Conf. Mobile Comput. Netw.*, 2013, pp. 75–86.

[12] T. W. Chim, S. M. Yiu, L. C. K. Hui, and V. O. K. Li, “VSPN: VANET-based secure and privacy-preserving navigation,” *IEEE Trans. Comput.*, vol. 63, no. 2, pp. 510–524, Feb. 2014.

[13] J. Ni, X. Lin, K. Zhang, and X. Shen, “Privacy-preserving real-time navigation system using vehicular crowdsourcing,” in *Proc. IEEE 84th Veh. Technol. Conf.*, 2016, pp. 1–5.

[14] J. Ni, K. Zhang, X. Lin, Y. Yu, and X. Shen, “Cloud-based privacy-preserving parking navigation through vehicular communications,” in *Proc. Int. Conf. Secur. Privacy Commun. Syst.*, 2016, pp. 85–103.

[15] K. Yang, K. Zhang, J. Ren, and X. Shen, “Security and privacy in mobile crowdsourcing networks challenges and opportunities,” *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 75–81, Aug. 2015.

[16] M. Gruteser and D. Grunwald, “Anonymous usage of location-based services through spatial and temporal cloaking,” in *Proc. 1st Int. Conf. Mobile Syst. Appl. Services*, 2003, pp. 31–42.

[17] R. Lu, X. Lin, H. Zhu, P. H. Ho, and X. Shen, “ECPP: Efficient conditional privacy preservation protocol for secure vehicular communications,” in *Proc. IEEE INFOCOM*, 2008, pp. 1903–1911.

[18] M. Li, F. Wu, G. Chen, L. Zhu, and Z. Zhang, “How to protect query and report privacy without sacrificing service quality in participatory sensing,” in *Proc. IEEE 34th Int. Perform. Comput. Commun. Conf.*, 2015, pp. 1–7.

[19] D. Pointcheval and O. Sanders, “Short randomizable signature,” in *Proc. RSA Conf. Topics Cryptology*, 2016, pp. 111–126.

[20] D. Chaum, “Blind signatures for untraceable payments,” in *Proc. Advances Cryptology*, 1983, pp. 199–203.

[21] S. Gisdakis, T. Giannetsos, and P. Papadimitratos, “SPPEAR: Security & privacy-preserving architecture for participatory-sensing applications,” in *Proc. ACM Conf. Secur. Privacy Wireless Mobile Netw.*, 2014, pp. 39–50.

[22] S. Rahaman, L. Cheng, D. F. Yao, H. Li, and J. -M. Park, “Provably secure anonymous-yet-accountable crowdsensing with scalable sublinear revocation,” *Proc. Privacy Enhancing Technol.*, vol. 2017, no. 4, pp. 384–403, 2017.

[23] J. Camenisch, J. M. Piveteau, and M. Stadler, “An efficient fair payment system,” in *Proc. ACM Conf. Comput. Commun. Secur.*, 1996, pp. 88–94.

[24] K. Wei, A. J. Smith, Y.-F. R. Chen, and B. Vo, “WhoPay: A scalable and anonymous payment system for peer-to-peer environments,” in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2006, p. 13.

[25] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system.” (2008). [Online]. Available: https://bitcoin.org/bitcoin.pdf

[26] D. Boneh and M. Franklin, “Identity-based encryption from the weil pairing,” in *Proc. Annu. Int. Cryptology Conf.*, 2001, pp. 213–229.

[27] C. P. Schnorr, “Efficient identification and signatures for smart cards,” in *Proc. Conf. Theory Appl. Cryptology*, 1989, pp. 239–252.

[28] (2018). [Online]. Available: https://www.travelchinaguide.com/cityguides/beijing/beijingmap.htm

[29] Mobile app released for digital payment of parking charges at all 1,100 public car parks in Singapore. (2017). [Online]. Available: https://www.opengovasia.com/articles /8082-mobile-app-released-for-digital-payment-of-parking-charges-at-all-1100-public-car-parks-in-singapore

[30] How to pay for parking. (2018). [Online]. Available: https://ottawa.ca/en/residents/transportation-and-parking/parking/how-pay-parking

[31] Parking guidebook for Chinese cities. Art. no. 38. (2014). [Online]. Available: https://www.itdp.org/w p-content/uploads/2014/07/Parking_Guidebook_for_Chinese_Cities.pdf

[32] (2018). [Online]. Available: https://www.shanghai-airport.com/parking.php

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

[34] T-Drive trajectory data sample, 2011. [Online]. Available: https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/

[35] M. Li, L. H. Zhu, Z. J. Zhang, and R. X. Xu, “Achieving differential privacy of trajectory data publishing in participatory sensing,” *Inf. Sci.*, vol. 400/401, pp. 1–13, 2017.

[36] M. Scott, “MIRACL: Multi-precision integer and rational arithmetic C/C++ Library.” (2018). [Online]. Available: http://www.certivox.com/miracl

[37] Z. Zhang, W. Cao, Z. Qin, L. Zhu, Z. Yu, and K. Ren, “When privacy meets economics: Enabling differentially-private battery-supported meter reporting in smart grid,” in *Proc. IEEE/ACM Int. Symp. Quality Service*, 2017, pp. 1–9.

[38] Tor, Orbot: Proxy with tor. (2018). [Online]. Available: https://guardianproject.info/apps/orbot/

[39] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, “SoK: Research perspectives and challenges for bitcoin and cryptocurrencies, in *Proc. IEEE Symp. Secur. Privacy*, 2015, pp. 104–121.

[40] (2018). [Online]. Available: https://www.thatsmyspot.com.au/. Accessed on: Jun. 19, 2018.

[41] M. Chiang, S. Ha, C.-L. I, F. Risso, and T. Zhang, “Clarifying fog computing and networking: 10 questions and answers,” *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 18–20, Apr. 2017.

[42] Z. Guan, J. Li, L. Wu, Y. Zhang, J. Wu, and X. Du, “Achieving efficient and secure data acquisition for cloud-supported Internet of Things in smart grid,” *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1934–1944, Dec. 2017.

[43] Z. Guan, G. Si, X. Zhang, L. Wu, N. Guizani, X. Du, and Y. Ma, “Privacy-preserving and efficient aggregation based on blockchain for power grid communications in smart communities,” *IEEE Commun. Mag.*, vol. 56, no. 7, pp. 1–7, 2018.

Liehuang Zhu is a professor with the School of Computer Science and Technology, Beijing Institute of Technology. He is selected into the Program for New Century Excellent Talents with the University from Ministry of Education, P. R. China. His research interests include cryptographic algorithms and secure protocols, Internet of Things security, cloud computing security, big data privacy, mobile and Internet security, and trusted computing. He is a member of the IEEE.

Meng Li is currently working toward the PhD degree in the School of Computer Science and Technology, Beijing Institute of Technology, and he is a visiting PhD student with Wilfrid Laurier University. He has received the National Graduate Student Scholarship in 2011 and he has been sponsored by China Scholarship Council in 2017. His research interests include applied cryptography, security and privacy, crowdsensing, fog computing, and blockchain. He is a student member of the IEEE.

Zijian Zhang is an associate professor with the School of Computer Science and Technology, Beijing Institute of Technology. He was a visiting scholar with the Computer Science and Engineering Department, State University of New York at Buffalo, in 2015. His research interests include cryptography, differential privacy, smart grid, data privacy, mobile security, and data mining. He is a member of the IEEE.

Zhan Qin is an assistant professor with the Department of Electrical and Computer Engineering, University of Texas at San Antonio. His research interests include secure computation outsourcing, privacy-preserving data collection, sharing and publication, cybersecurity of smart grid control and communication system, and cyber-physical security for smart devices with the current focus on exploring and improving the security and privacy assurance on cloud computing. He is a member of the IEEE.