This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

Towards Privacy-preserving Valet Parking in Autonomous Driving Era

Jianbing Ni, *Member, IEEE,* Xiaodong Lin, *Fellow, IEEE,* Xuemin (Sherman) Shen, *Fellow, IEEE*

***Abstract*—Automated valet parking, deemed as a key milestone on the way to autonomous driving, has great potential to improve the "last-mile" driving experience for users. On the other hand, it triggers serious risks on vehicle theft and location privacy leakage. To address these issues, we propose a secure and privacy-preserving automated valet parking protocol for self-driving vehicles. The proposed protocol is characterized by extending anonymous authentication to support two-factor authentication with mutual traceability for reducing the risks of vehicle theft and preventing the privacy leakage of users in automated valet parking. Specifically, based on one-time password and secure mobile devices, two-factor authentication is achieved between vehicles and smartphones to ensure vehicle security in remote pickup. By exploiting the BBS+ signature and the Cuckoo filter, user location privacy is protected against the curious parking lots and service providers. In addition, the traceable tags are designed to enable a trusted authority to identify the vehicles and users for localizing a stolen or missing vehicle and preventing the slandering of greedy users. Finally, formal security analysis on the proposed protocol is given to show that the authentication, anonymity and traceability can be reduced to standard hard assumptions, and performance evaluation demonstrates the proposed protocol is efficient and practical to be implemented in autonomous driving era.**

Keywords: Autonomous vehicles, automated valet parking, remote control, privacy preservation.

I. Introduction

Over the past few years, the automotive industry and technology corporations (e.g., Google, Uber, and Baidu) have made significant leaps to render self-driving vehicles a reality [1]. Equipped with advanced sensing and communication capabilities and intelligent control systems, autonomous vehicles (AV) are capable of identifying relevant signals and obstacles, analyzing traffic conditions and generating appropriate navigation paths to deliver passengers to destinations without any intervention from humans. They have the great potential to fundamentally alter transportation systems by increasing personal safety, enhancing user satisfaction, decreasing environmental interruption, reducing infrastructure cost, and saving time for drivers [2]. Despite the appealing benefits that attract great efforts from both global automakers and technology companies, there is still a long way to go before customization and regulations for fully autonomous vehicles.

Although AVs are not commercially available now, rapid developments are creating a series of use cases for embracing the autonomous driving era, one of which is automated valet parking (AVP) [3], [4]. AVP allows users to leave their vehicles at the drop-off area, e.g., the foyer of a hotel and the departure layer of the airport, and activate the AVP application on the smartphone for valet parking. AVs connect with the nearby parking lots to find vacant parking space, self-drive to the parking lots, and park themselves at the vacant space. Due to the development of advanced sensing, communication and control technologies, parking space discovery and vehicle parking are completely automatic and problem-free for AVs. When departure, users activate the AVP application, and AVs start the engines and self-drive to the pickup positions, and users can jump in and start their trips. AVP frees users from the troublesome vehicle parking and pickup issues, including time and gas costs for vacant parking search, wandering around parking garages for vehicle finding, and unpleasant experience when walking to parking areas, and enables users to have vehicles at anytime and from anywhere they need. For example, users can avoid the situations where vehicles are needed at workplace but they are parking at home. AVP is deemed as a key milestone on the way to autonomous driving. Currently, Mercedes-Benz [4] has provided the first infrastructure-assisted solution for AVP service to allow users to drop-off and pick-up their vehicles at fixed positions. Tesla [5] is also promoting the self-park service at parking lots without drivers' intervention and GM [6] has patented the automatic valet parking by proposing a method of autonomously retrieving a vehicle using a wireless device.

While the emerging AVP receives extensive attentions from automobile industry, it also brings new and challenging security and privacy threats towards both AVs and users. First of all, due to the exposure of remote control interface, AV is facing with a broad range of cyber attacks [7], which may result in the safety issues to drivers, passengers and pedestrian [8]. The success of malicious remote control over smart vehicles has been exhibited in the experiment of remotely killing a Jeep on the highway [9]. Secondly, AVP requires users to remotely start engines, which significantly increases risks of vehicle theft as remote starters help thief bypass factory security systems. Thirdly, AVP service providers are separate administrative entities maintaining all users' identity information and trajectories, and parking and pickup transcripts exchanged between smartphones and remote AVs. As a result, user privacy is being put at risk due to the curiosity of AVP service providers. Once accessing AVP services, users actually relinquish the ultimate protection on their locations. In short, although AVP is intuitively attractive for users, it does not immediately provide sufficient security and privacy guarantees.

Jianbing Ni and Xuemin (Sherman) Shen are with Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1, email: {j25ni, sshen}@uwaterloo.ca.

Xiaodong Lin is with School of Computer Science, University of Guelph, Guelph, Ontario, Canada N1G 2W1, email: xlin08@uoguelph.ca.

Corresponding Author: Xiaodong Lin.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

This issue, if not resolved properly, may impede the success of AVP services.

To secure vehicle control, single-factor authentication based on physical key or biometric is widely implemented on vehicles, which results in endless vehicle theft accidents nowadays. Traditional physical keys or fingerprints can be copied easily and the popular passive keyless entry and start systems have been proved to be vulnerable to relay attack [10] and correlation-based attack [11]. Remote vehicle control offers large automotive attack surfaces to adversaries, such that identity authentication with higher security guarantee is necessary for AVs. Multi-factor authentication with the integration of multiple authentication factors, e.g., secret data, secure devices, biometric and password, is a promising candidate for identity verification between smartphones and AVs. n-factor authentication upgrades the security of single-factor authentication that even any n-1 factors are disclosed accidently, security guarantee would not be degenerated. Due to this appealing property, many two/three factor authentication schemes and its variant, e.g., anonymous or mutual authentication, have been proposed to secure different scenarios, e.g., wireless sensor networks [12], [13], mobile network [14], vehicular ad hoc networks [15], and e-healthcare [16]. However, because of the constrained computing capability of mobile phones and AVs, the computational efficiency is an essential factor deserves to be taken into account in designing n-factor authentication suitable for secure authentication between smart phones and AVs.

To preserve user privacy, anonymity technique is important for user's identity hidden during AVP service access. As AVP service providers and other entities cannot acquire any knowledge about users, it is impossible to prevent misbehavior from greedy or dishonest users. For example, a greedy user can claim vehicle loss accidents after he/she retrieves the AV, or accuse an honest user of the scratching of his/her AV in the public parking lot. Although the parking lot might be irresponsible to compensate the user for vehicle loss, it has to be engaged in the investigation. To prevent slandering attacks, perfect anonymity is not recommended for AVP systems. The traceability is of equal importance to recover a user's identity when necessary for a trusted third party. In addition, perfect privacy preservation brings troubles on the tracing of lost vehicles, which is essential for users or police to localize lost vehicles. Another challenging problem is the loss of smartphones. If the user loses the smartphone, he/she cannot retrieve the vehicle when needed. More seriously, the user is not able to use the physical key to drive the vehicle for the lack of the vehicle's location. Therefore, for the success of AVP services, it is desirable to take into account the aforementioned critical issues, including the tracing of anonymous misbehaving users and the concerns of vehicle and smartphones loss.

To promote the AVP services, we propose a privacy-preserving automated valet parking protocol (PrivAV) for AVs. PrivAV is characterized by supporting two-factor authentication and mutual traceability for preventing vehicle theft, while achieving anonymous authentication to protect user location privacy in automated valet parking. More specifically, not only does PrivAV enable two-factor authentication between smartphones and AVs for securing remote retrieval, but support anonymous AVP service access without exposing sensitive information about users. Further, a trusted third party (i.e., a judge) is engaged in the AVP service to trace the anonymous users and lost AVs for preventing slandering and maintaining fairness. Specifically, the main contributions of this paper are three folds as follows:

- A secure automated valet parking protocol is proposed to prevent vehicles from being hacked or stolen. To ensure secure remote control of AVs, two-factor authentication is introduced based on one-time password and smart device to resist malicious access on AVs. As a result, an attacker that has either one-time password or smartphone is unable to start the engine and drive the vehicle away.
- Anonymous service authentication between users and AVP servers and location privacy preservation for users are achieved based on the BBS+ signature [17] and the Cuckoo filter [18]. PrivAV strengthens AVP services by assisting users to build secure remote connections with their AVs and protecting all transcripts between users and AVs without invading users' privacy.
- An honest judge is involved to trace the anonymous users for refraining misbehavior during vehicle retrieval, and recover the locations of AVs to help police find lost vehicles under the delegation of AV owners. In addition, the judge can recover the parking garage where an AV is parking for the user in case his/her smartphone is lost.
- Security analysis is given to show that the authentication, anonymity and traceability of PrivAV rely on the standard hard assumptions, including CDH, $q-$SDH and DBDH assumptions, and performance evaluation demonstrates PrivAV is efficient to be implemented in reality.

The remainder of this paper is organised as follows. In section II, we present the AVP architecture and threat models, and identify design objectives. In section III, we review preliminaries. Then, we propose our PrivAV in section IV and prove its security in section V, followed by the efficiency analysis in section VI. Finally, we review related work in section VII and draw the conclusion in section VIII.

II. Problem Statement

In this section, we present the AVP architecture, threat model and identify design objectives.

A. The Entities

An AVP system is composed of five entities: an AVP service provider, parking lots, users, AVs, and the judge. The AVP service provider manages AVP servers to provide automated parking and pickup services to users. It is responsible to solve the problems brought from stolen vehicles and lost smartphones. The parking lots have deployed cameras, communication units, and a local server to manage the status of a parking space, i.e., vacant, reserved or possessed, on behalf of a fog node [19]. A user has a smartphone, which is used to remotely control her/his AV for parking and pickup. Each user needs to register at the AVP server to obtain a service credential. The autonomous vehicle has a temper-proof

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

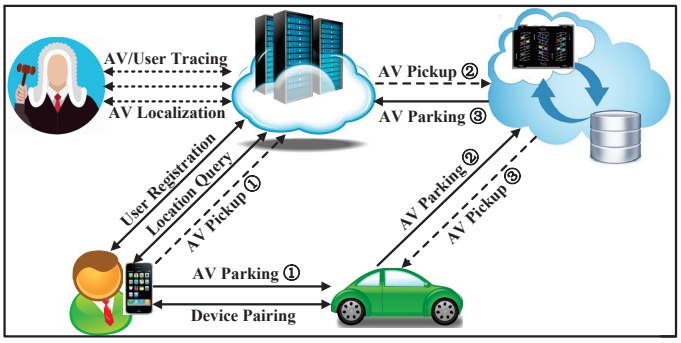


Fig. 1. AVP Architecture

on-board unit to communicate with the user's smartphone, roadside units and communication units in the parking lot. It has the remote control interface provided to its owner for parking and pickup after the smartphone and the AV are paired. The judge is a trusted third party, who is the police or a government agency that administers vehicle registration and driver licensing. The judge has the public information about all the entities, including the system parameters of AVP services, public keys of participating entities, parking lot information, and vehicle registration information. It has the power to trace all the AVs and investigate the event of vehicle theft.

B. Overall Architecture

We briefly describe the overall AVP architecture. We assume that each smartphone and on-board unit (OBU) have a small amount of read-only memory, which is initialized during registration. In our PrivAV, the read-only memory will be used to maintain user identity and secret values generated during service setup and user registration, i.e., secret keys and service credentials. The device can freely read the contents and modify its memory except the read-only memory.

The scenario of automated valet parking is shown in Fig. 1, and the overall architecture consists of the protocols as described below.

1) *Service Setup*. The AVP service provider setups the whole AVP service and chooses the system parameters. The system parameters are public, as well as the public keys of all the entities. The secret keys are managed by their owners secretly.
2) *Device Pairing*. A smartphone and an AV owned by a user make a pair by exchanging their public keys and random values. At the end of the protocol, a SP-AV key pair is initialized and shared between the smartphone and AV. In practice, this process may be carried out when the user purchases the AV from the dealer. The smartphone and the OBU on the AV may interact through device-to-device communications supported by Boothtooth or Wi-Fi.
3) *User Registration*. The user contacts the service provider for registration on AVP servers by committing the secret key. At the end of the protocol, the user acquires the service credential, which is maintained on the read-only memory of the smartphone. In practise, this process may be carried out when the user desires to access AVP services. The smartphone may communicate with the AVP servers through Wi-Fi or cellular networks.
4) *AV Parking*. The OBU carries an interactive protocol with the smartphone, the AVP server and the local server in the parking lot. Under the user control with the smartphone, the AV starts the automated parking mode to find an accessible parking lot automatically and navigate to arrive the vacant parking space using autonomous driving after the user leaves it at the drop-off place. In this process, the user initializes the one-time password and the pickup code with the AV that would be utilized for AV pickup when departure. At the end of this protocol, the AVP server acquires the protected location of vehicles and other auxiliary information for vehicle tracing. In practice, the OBU and the smartphone can communicate through Boothtooth or Wi-Fi, the OBU and the smartphone can communicate with the AVP server and the local server in the parking lot via Wi-Fi or cellular network, and the other entities, such as cameras, roadside-units, local servers and AVP servers, can be connected through direct cable connection.
5) *Location Query*. The smartphone carries an interactive protocol with the AVP server. This protocol takes the pickup code as inputs and returns the AV's location to the smartphone.
6) *AV Pickup*. The smartphone carries an interactive protocol with the remote AV through the AVP server and the local server in the parking lot. This protocol takes its SP-AV key pair, one-time password, pick-up code, the location and time where and when the user catches the AV as input, and outputs the authentication result from the AV. It also enables the user to remotely pay parking fee through the smartphone based on anonymous electronic cash or bitcoin systems. At the end of the protocol, the AV starts the engine and cruises to the pickup point to pick the user up using the autonomous driving technique, if the pickup request is from a correct user.
7) *User Tracing*. The judge can trace the user of an AV if he/she has performed some illegal activities. Given the transcripts maintained on AVP servers, the judger can trace the owner of a particular AV in case that the AV is accused to scratch with other vehicles, a stolen AV is found by the police or a user claimed the loss of his/her AV but actually he/she has pick it up.
8) *AV Tracing*. The judge can seek the consent from the user to trace the parking lots. Given the user's consent, the judge is able to trace all the parking lots where the AV has parked, including those in the future (in the case of stolen AV tracing).
9) *AV Localization*. The user can have the parking lot of the AV with the assistance of the judge. The judge is able to recover the parking lot of a specific AV and the user can pick it up using the physical AV key in case of smartphone loss.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

C. Threat Model and Design Objectives

We consider that the entities in the architecture honestly follow the specified rules in AVP services, for example, the service provider can execute the protocols agreed with users, and the users would not violate the consented policies during service access. Thus, the users trust the AVP service providers and prefer to use the offered AVP services. However, the AVP service is confronted with a variety of attacks from outsiders and insiders or the collusion of both. A concrete example is that the adversary may eavesdrop on communication channels in an attempt to breach location privacy of an honest user. The local server, whose responsibility is to manage the parking lot, is also curious to the privacy of users. However, we do not consider the privacy leakage though cameras. In this paper, we consider the following four kinds of attacks, namely, AV stealing attacks, location privacy infringement, slandering and hiding, which are detailed as follows.

- *AV Stealing Attacks*. The attacker tries to steal users' AVs in the parking lots through cyber attacks, including impersonation attack, replay attack, forging attack and man-in-the-middle attack. The attacker could be either outsider or insider who remotely steals a particular AV. The objective is to hack the authentication between smartphones and AVs, and thereby start the engine and drive the AVs away. Else, we consider the case that an attacker may accidently pick up or maliciously steal the smartphone of a specified user and aims to steal his/her AV by using the smartphone on which the scream lock password does not set or has been broken. The traditional approaches, such as towing vehicles, using skeleton keys or electronic interference, and physical attacks, do not covered in this paper, these misbehavior can be recorded by the cameras deployed in parking lots.
- *Location Privacy Infringement*. The attacker tries to identify the location of a particular user. We consider a powerful adversary that can collude with other users and access the local servers and AVP servers. Given the secret keys of these entities, the attacker's goal is to decide if a particular message is sent from one of the two honest users, so as to predict the user's location based on the AV's position. To guarantee strong privacy preservation, we enhance the threat model that even if an attacker physically follows a certain AV to eavesdrop all its received massages, and has accessed to the secret keys of the local server and AVP server, the attacker still cannot corrupt location privacy of an honest user, if it does not know the owner of this AV in advance. We does not consider the ownership exposure of a particular AV through other ways, such as physical observation or cameras. If the ownership is known by the attacker, it is obvious that the location privacy is no longer preserved. Therefore, our goal is to preserve the location privacy of users even the attacker is able to access all the information exchanged between all entities in the architecture during parking and pickup processes, except those exchanged between the smartphone and the OBU in device pairing, as long as the attacker does not know the AV's owner.
- *Slandering*. The attacker tries to slander an honest user after the AV is scratched. It could be a registered but malicious user who releases a piece of exchanged information to the judge, such that the judge would identify an honest user is the scratching perpetrator. Although the cameras in parking lots may record the event, it is hard to ensure all the accidents are captured in the parking lots. Another slandering is that a registered but greedy user may slander the parking lots for the AV loss but in fact he/she has pick it up. Due to the privacy preservation of users, the parking lots cannot determine who picked the AV up, such that it is possible for a user to slander the parking lot for AV lost. Although the parking lots can claim the irresponsibility of vehicle loss to avoid the compensation, they may be required to provide the videos and witnesses for investigation.
- *Hiding*. The insiders, such as users and parking lots, or AV thief may have different incentives to hide the misbehavior. Firstly, the attacker, who is a registered user, tries to conduct a pickup message that cannot be traced by the judge. Thus, the attacker can claim the AV loss event to slander the parking lot. A registered user may also try to keep anonymous for escaping from compensation if the AV is involved in a scratch or traffic accident. Secondly, the parking lot is willing to be invisible in an attempt to avoid troubles during accident investigation. Finally, an attacker who stoles an AV may try to prevent the judge and the user from localizing the AV.

We aim to propose an automated parking and pickup architecture that is secure against the given threats and privacy-preserving for users. To prevent slandering and hiding, PrivAV should support correct tracing of users and AVs. The parking lots where the AV parked should be recovered when the judge is involved to investigate the vehicle loss.

III. Preliminaries

We briefly review several underlying techniques employed to construct PrivAV, including bilinear pairing, BBS+ signature, zero-knowledge proof and Cuckoo filter.

Bilinear Pairing. Let $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ be three cyclic groups of the same prime order p. $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a type-3 bilinear map [20] if the following properties are satisfied:

- (Bilinearity). For any $g \in \mathbb{G}_1, h \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$.
- (Non-Degeneracy). For any $g \in \mathbb{G}_1 \setminus \{1_{\mathbb{G}_1}\}$, $h \in \mathbb{G}_2 \setminus \{1_{\mathbb{G}_2}\}$, $\hat{e}(g, h) \neq 1_{\mathbb{G}_T}$.
- (Efficient Computability). For any $g \in \mathbb{G}_1, h \in \mathbb{G}_2$, $\hat{e}(g, h)$ is efficiently computable.
- (Non-Homomorphism). No efficiently computable homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$ exists in either direction.

Mathematical Assumptions. We review three cryptographic assumptions related to the security of PrivAV.

Computational Diffie-Hellman [21]. The Comutational Diffie-Hellman (CDH) problem in $\mathbb{G}_1$ is defined as follows: Given a tuple $(g, g^a, g^b) \in \mathbb{G}_1^3$, to compute g^{ab}. We say that the CDH assumption in $\mathbb{G}_1$ holds if there is no algorithm can solve the CDH problem in $\mathbb{G}_1$ with non-negligible advantage

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

in probabilistic polynomial time.

q-Strong Diffie-Hellman [22]. The q-Strong Diffie-Hellman (q-SDH) problem is defined as follows: Given a $(q+2)$ tuple $(g, g_0, g_0^x, g_0^{x^2}, \cdots, g_0^{x^q}) \in \mathbb{G}_1^{q+2}$, output a pair (A, c) such that $A^{x+c} = g_0$, where $c \in \mathbb{Z}_p$. We say that the q-SDH assumption holds if there is no algorithm can solve the q-SDH problem with non-negligible advantage in probabilistic polynomial time.

Decisional Bilinear Diffie-Hellman [21]. The Decisional Bilinear Diffie-Hellman (DBDH) problem is defined as follows: Given a tuple $(g_i, g_j, g_k, g_i^a, g_j^b, g_k^c)$ and $\hat{e}(g_1, g_2)^z$, where $i, j, k \in \{1, 2\}$, to determine whether $\hat{e}(g_1, g_2)^{abc} = \hat{e}(g_1, g_2)^z$. We say that the DBDH assumption holds if there is no algorithm can solve the DBDH problem with non-negligible advantage in probabilistic polynomial time.

BBS+ Signature. The BBS+ signature is proposed by Au et al. [17] based on the BBS signature [23] and the CL signature [24]. Let g, g_0, g_1, g_2, g_3 be the generators of $\mathbb{G}_1$, h be the generator of $\mathbb{G}_2$, and the signer's secret key is $\alpha \in \mathbb{Z}_p$ and the corresponding public key is $\beta = h^\alpha$. The signature over a tuple of messages (m_1, m_2, m_3) is (A, e, s), in which $A = (gg_0^s g_1^{m_1} g_2^{m_2} g_3^{m_3})^{\frac{1}{\alpha+e}}$ and (e, s) are randomly chosen from $\mathbb{Z}_p$. The signature is verified by checking whether $\hat{e}(A, \beta h^e) = \hat{e}(gg_0^s g_1^{m_1} g_2^{m_2} g_3^{m_3}, h)$ holds or not. The BBS+ signature allows the signer to generate the signature in a blind way, i.e., the signer signs a commitment on a tuple of messages (m_1, m_2, m_3) without having the messages. The security of BBS+ signature is proved under the q-SDH assumption against adaptive chosen message attacks [17].

Zero-knowledge Proof. In a zero-knowledge-proof (ZoK) protocol [25], a prover is able to convince a verifier that he knows a witness w satisfying some relation R with respect to a known string x without revealing anything other than the relation $(w, x) \in R$. Currently, a plethora of ZoK protocol have been proposed, in which Σ-protocol is a special kind of three-move non-interactive ZoK protocol with honest verifiers. The Σ-protocol that the prover has an integer x so that the relation $y = g^x$ holds is denoted as $\mathcal{PK}\{(x) : y = g^x\}$. The values in the parentheses denote the knowledge that the prover aims to prove, and the values on the right are the publicly known values. Besides, the Σ-protocol can be transformed into a non-interactive signature proof-of-knowledge (SPK) protocol. The SPK protocol transformed from the above ZoK protocol is denoted as $SPK\{(x) : y = g^x\}(m)$.

Cuckoo Filter. The cuckoo filter is designed by Fan et al. [18] that can replace Bloom filter for set membership tests. Compared with the standard Bloom filter, cuckoo filter not only supports item adding and removing dynamically (the standard Bloom filter cannot provide item deletion), but also has high lookup performance, even when close to full. In addition, the cuckoo filter is easy to be implemented and possesses less storage space than Bloom filter in applications, when the false positive rate is less than 3%.

A cuckoo filter is a compact variant of a cuckoo hash table which consists of an array of buckets in which each item has two candidate buckets determined by two hash functions $H_1(x)$ and $H_2(x)$. To lookup an item x, one can check both buckets to see whether either contains this item. Cuckoo hashing reaches high space occupancy since it can relocate the earlier-added items if both buckets are possessed when a new item inserts. For example, Fig. 2 shows that an item x can be inserted to either budget 3 or 7 in a hash table of 8 buckets, if either of x's buckets is empty; otherwise, x will be put in one of the buckets, kick out the existing item (if c) and re-inserts c to its alternate location. The cuckoo hashing can be used to provide set membership information. Cuckoo filter is designed to improve hash table performance by using partial-key cuckoo hashing. For an item x, the hashing scheme used to generate the indexes of two buckets is as follows: $H_1(x) = hash(x)$ and $H_2(x) = H_1(x) \oplus hash(x's\ fingerprint)$. The fingerprint of x is maintained on one bucket with the index $H_1(x)$ or $H_2(x)$, if either is not possessed; Otherwise, the item y maintained on the bucket $H_1(x)$ or $H_2(x)$ is kicked out and kept on the bucket $j \oplus hash(y's\ fingerprint)$, where j is $H_1(x)$ or $H_2(x)$. To test the item x, the lookup algorithm calculates $x's\ fingerprint$ and $H_1(x) = hash(x)$, $H_2(x) = H_1(x) \oplus hash(x's\ fingerprint)$. If $x's\ fingerprint$ is in either bucket $H_1(x)$ or $H_2(x)$, the cuckoo filter returns true; Otherwise, it returns false. To remove an item from the entire filter, the delete algorithm computes $H_1(x)$ and $H_2(x)$, finds $x's\ fingerprint$ in $H_1(x)$ or $H_2(x)$ and deletes that value.

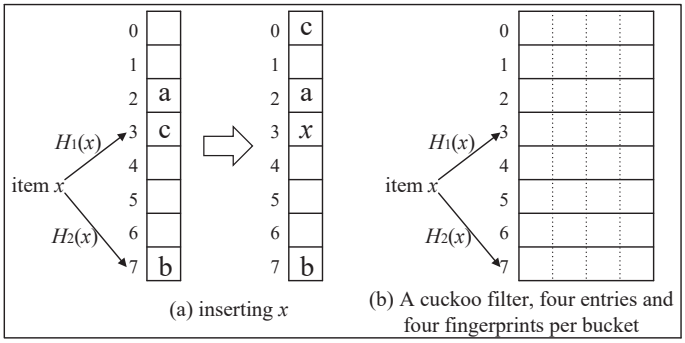


Fig. 2. Cuckoo Hash Table and Cuckoo Filter [18]

IV. Our Proposed PrivAV

In this section, we describe the proposed PrivAV and discuss its properties.

A. The Detailed PrivAV

Service Setup. The service provider bootstraps the whole AVP service on the servers. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a type-3 bilinear pairing as discussed in III, $g, g_0, g_1, g_2, g_3, g_4$ be the generators of $\mathbb{G}_1$ and $h, h_0, h_1, h_2, h_3, h_4$ be the generators of $\mathbb{G}_2$. The AVP server picks $\alpha \in_R \mathbb{Z}_p$ as its secret key and computes the corresponding public key as $\beta = h^\alpha$, where $\alpha \in_R \mathbb{Z}_p$ denotes that α is randomly chosen from $\mathbb{Z}_p$. $H : \{0,1\}^* \to \mathbb{Z}_p$ and $H_g : \{0,1\}^* \to \mathbb{G}_1$ are two collision-resistant hash functions. $G_0 = \hat{e}(g_0, h)$, $G_1 = \hat{e}(g_1, h)$, $G_2 = \hat{e}(g_2, h)$, $G_3 = \hat{e}(g_3, h)$, $G_4 = \hat{e}(g_4, h)$. In addition, the AVP server builds a cuckoo filter CF_P for the maintenance of pickup codes. The system parameter is $Param = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, g, g_0, g_1, g_2, g_3, g_4, h, h_0, h_1, h_2, h_3, h_4, H, H_g, G_0, G_1, G_2, G_3, G_4, \hat{e}, \beta)$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

The judge setups the secret-public key pair (t, T), in which $t \in_R \mathbb{Z}_p$ and $T = h^t \in \mathbb{G}_2$.

A parking lot with its identifier $\mathcal{L} \in_R \mathbb{Z}_p$ has a secret-public key pair $(l, L = g^l)$, in which $l \in_R \mathbb{Z}_p$ and $L = h^l \in \mathbb{G}_2$. (l, L) is stored on the local server of the parking lot.

Device Pairing. A user with the identity $I \in \mathbb{Z}_p$ has the secret key pair (u, U) maintained on the read-only memory of the smartphone, in which $u \in_R \mathbb{Z}_p$ and $U = g^u \in \mathbb{G}_1$. Suppose the user purchases a new AV from a dealer, whose licence plate number is $N \in \mathbb{Z}_p$. The secret key of the AV is $v \in_R \mathbb{Z}_p$ and the corresponding public key is $V = h^v \in \mathbb{G}_2$. To pair the smartphone and AV, the smartphone chooses $u_1 \in \mathbb{Z}_p$ to compute $U_1 = h^{u_1}$ and sends (I, U, U_1, T_u) to the AV; The OBU on the AV randomly selects $v_1 \in \mathbb{Z}_p$ to compute $V_1 = g^{v_1}$ and sends (N, V, V_1, T_v) to the smartphone, where T_u, T_v are timestamps. The smartphone and the AV separately generate $w = H(U^I, V^N, U_1^v, V_1^u, T_u, T_v)$, $W = g^w$ and thereby confirm the same (w, W) based on the secure Station-to-Station (STS) protocol [26]. Thus, the SP-AV secret-public key pair is (w, W).

User Registration. The user is required to register at the AVP server to acquire a service credential, which is used to access AVP without disclosing the identity information. The user engages with the AVP server through a secure channel and enrolls into the service as follows.

- The user randomly chooses $s', r' \in_R \mathbb{Z}_p^2$ to compute a commitment $C = g_0^{s'} g_1^u g_2^w g_4^{r'}$, and sends (C, I, W) to the AVP server, along with the following proof:

$$\mathcal{PK}_1\{(s', u, w, r') : C = g_0^{s'} g_1^u g_2^w g_4^{r'} \wedge W = g^w\}.$$

  $\mathcal{PK}_1$ assures that C is computed correctly.
- The AVP server verifies the validity of $\mathcal{PK}_1$ and randomly chooses $s'', e, r'' \in_R \mathbb{Z}_p^3$ to calculate $A = (C g g_0^{s''} g_3^I g_4^{r''})^{\frac{1}{\alpha+e}}$, and returns (A, s'', e, r'') to the user. The AVP server stores $(C, A, s'', I, e, r'', W)$ secretly.
- The user calculates $s = s' + s''$, $r = r' + r''$, and checks whether $\hat{e}(A, \beta h^e) \stackrel{?}{=} \hat{e}(g g_0^s g_1^u g_2^w g_3^I g_4^r, h)$. Note that $\sigma_s = (A, e, s)$ is the BBS+ signature on (I, u, w, r), where r remains unknown to the AVP server. The tuple (A, e, s, I, r, s', r') is maintained on the smartphone and (I, r) is stored in the read-only memory.

AV Parking. The user arrives the destination and leaves the AV at the get-off point, such as the foyer of a hotel or the departure layer of the airport. The user uses the smartphone to initialize a one-time password pwd for pickup authentication and a pickup code $code \in_R \mathbb{Z}_p$ for vehicle search, and sends $(pwd, code)$ to the OBU. The communication between the OBU and the smartphone is secure by utilizing their shared SP-AV key pair (w, W). pwd is set by the user. $code$ is randomly chosen from $\mathbb{Z}_p$, and stored on the smartphone and the OBU. Then, the user sends a parking command to the AV and the latter starts the automated parking mode to automatically discovery a vacant parking space based on real-time parking information maintained on automated parking navigation systems.

After arriving a vacant parking space in a parking lot with the secret-public key pair (l, L), the AV interacts with the local server to update the parking information. Specifically, the AV performs the following operations to generate a parking message $\mathcal{P}$ and sends it to the local server.

- Encrypt its current location AV_L using w based on the advanced data encryption (AES), and encrypt the pickup code $code$ using β based on the ElGamal scheme to obtain $(E_w(AV_L), E_\beta(code))$, respectively;
- Select $\gamma \in_R \mathbb{Z}_p$ to compute a commitment $C' = g^\gamma g_1^v g_2^w g_3^N$ and $R = T^\gamma$;
- Compute a searchable tag $S = \hat{e}(H_g(I, N, U)^I, h^\gamma)$;
- Generate the zero-knowledge proof:

$$\mathcal{PK}_2\{(w, v, N, \gamma) : C' = g^\gamma g_1^v g_2^w g_3^N \wedge \ R = T^\gamma\};$$

- Set $\mathcal{P} = (\mathcal{L}, E_w(AV_L), E_\beta(code), C', R, S, \mathcal{PK}_2)$.

Upon receiving $\mathcal{P}$ from an AV, the local server in the parking lot verifies the validity of $\mathcal{PK}_2$ and generates a signature on C' by choosing $\gamma', b \in_R \mathbb{Z}_p^2$ to compute $B = (g_0 C' g_4^{\gamma'})^{\frac{1}{l+b}}$ and $R' = h^{\gamma'}$. At last, the local server forwards $(\mathcal{P}, B, b, R')$ to the AVP server.

The AVP server checks the validity of $\mathcal{PK}_2$ and decrypts $E_\beta(code)$ using the secret key α to obtain $code$, which enables the server to search the corresponding AV. Then, the AVP server computes the tuple $H_1(code) = hash(code)$, $H_2 = H_1(code) \oplus hash(code's\ fingerprint)$, and inserts it to the cuckoo filter CF_P. Finally, the AVP maintains the parking message $\mathcal{P}$ securely on the memory.

Location Query. If the user is willing to query the location of his/her AV, the smartphone randomly chooses $\mu_1 \in_R \mathbb{Z}_p$ to compute $\psi_1 = h^{\mu_1}$ and encrypts $code$ using β based on the ElGamal scheme to acquire $E_\beta(code)$. Then, the smartphone sends ("Q", $E_\beta(code), \psi_1$) to the AVP server, where "Q" denotes that this message is to query an AV's location. Upon receiving ("Q", $E_\beta(code), \psi_1$), the AVP server decrypts $E_\beta(code)$ using α to obtain $code$ and performs the lookup algorithm to test whether $code$ exists in CF_P. If yes, it selects $\mu_2 \in_R \mathbb{Z}_p$ to compute $\psi_2 = h^{\mu_2}$ and $\psi = \psi_1^{\mu_2}$, and returns $(E_{\psi_1}(AV_L), \psi_2)$ to the user. Then, the AVP server executes the delete algorithm to remove $code$ from CF_P, computes $code' = H(code, \psi)$ and inserts $code'$ to CF_P. Finally, the smartphone updates $code$ as $code' = H(code, \psi_2^{\mu_1})$ and uses μ_1 to recover the AV's location AV_L from $E_{\psi_1}(AV_L)$. This protocol can be performed more than once for enabling the user to frequently check the AV's location.

AV Pickup. Before departure, the user inputs the one-time password pwd and the location and time $(U_L, Time)$ where and when he/she is willing to be picked up. The user sends the remote AV a pickup request $\mathcal{D}$, which is generated by the smartphone as follows.

- Encrypt $(U_L, Time)$ using the shared SP-AV key w based on the AES scheme to acquire $E_w(U_L, Time)$, and encrypt $code'$ with β to obtain $E_\beta(code')$;
- Randomly select τ to compute $F = h^\tau$ and $G = G_0^s G_1^u G_2^w G_4^r \hat{e}(g^\tau, T)$ and $J = (g_0^{H(pwd||0)} g_1^{H(pwd||1)})^\tau g_2^w$;
- Generate the following non-interactive zero-knowledge

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

proof from the service credential (A, e, s).

$$\mathcal{SPK}\left\{\begin{array}{l}(A,e,s,u,w,I,r,\tau):\\ \quad \hat{e}(A,\beta h^e)\stackrel{?}{=}\hat{e}(gg_0^s g_1^u g_2^w g_3^I g_4^r,h)\ \wedge\\ \quad\quad F=h^\tau\ \wedge\\ \quad\quad G=G_0^s G_1^u G_2^w G_4^r \hat{e}(g^\tau,T)\ \wedge\\ \quad\quad J=(g_0^{H(pwd||0)}g_1^{H(pwd||1)})^\tau g_2^w\end{array}\right\}(P);$$

where "P" denotes this is a pickup request.

- Set $\mathcal{D}=(\text{"}P\text{"}, E_\beta(code'), E_w(U_L, Time), F, G, J, \mathcal{SPK})$.

Note that the payment information of parking fee can be added into the pickup request $\mathcal{D}$. To ensure the privacy preservation, the user can choose anonymous payment methods, such as electronic cash or bitcoin.

Upon receiving $\mathcal{D}$ from the smartphone, the AVP server verifies the validity of $\mathcal{SPK}$. If it is invalid, it returns failure and aborts; Otherwise, the AVP server decrypts $E_\beta(code')$ to recover $code'$, and performs the lookup algorithm of cuckoo filter to check whether $code'$ is present in CF_P. If yes, the AVP server finds the corresponding parking lot $\mathcal{L}$ based on $\mathcal{P}$, and forwards $(\text{"}P\text{"}, code', E_w(U_L, Time), J, \psi)$ to the local server of $\mathcal{L}$. The AVP server maintains $\mathcal{D}$ in the memory.

The local server verifies $\mathcal{SPK}$ and broadcasts a message $(code', \psi)$ to all the AVs in the parking lot $\mathcal{L}$.

Each AV uses the maintained pickup code $code$ to check whether $code' \stackrel{?}{=} H(code, \psi)$. If not, the AV aborts; Otherwise, it returns $code$ to the local server. The local server checks the parking payment information and forwards $(E_w(U_L, Time), J)$ to the AV if payment is successful. The AV decrypts $E_w(U_L, Time)$ to acquire the pickup position and time of the user. The AV also reads (pwd, w) from its memory and verifies whether $\hat{e}(Jg_2^{-w}, h) \stackrel{?}{=} \hat{e}(g_0^{H(pwd||0)} g_1^{H(pwd||1)}, F)$. If yes, the AV returns an ACK message to the user through both servers, starts its engine and autonomously drives to the defined position U_L at $Time$ to pick up the user. Otherwise, the AV returns failure and drops the pickup request. Upon receiving ACK from the AV, the AVP server can delete $code'$ from the cuckoo filter CF_P.

User Tracing. To trace the user who picked up the AV N, the judge retrieves (F, G) in the pickup message $\mathcal{D}$ and all the registration transcripts $(C, A, s'', I, e, r'', W)$ from the AVP server. The judge leverages its secret key t to compute $G' = G\hat{e}(g, F)^{-t}$ and tests whether $G' \stackrel{?}{=} \hat{e}(C, h)G_0^{s''}G_4^{r''}$ until finding a match. Additionally, the judge is able to know the AV N has been picked up by the user I.

AV Tracing. To trace the AV during parking, the user provides (I, U, h^w) to the judge and the judge can retrieve all the parking messages $\mathcal{P}$ from the AVP server. The judge tests $\hat{e}(C', h) \stackrel{?}{=} \hat{e}(g, R)^{-t}\hat{e}(g_1, V)\hat{e}(g_2, h^w)\hat{e}(g_3, h^N)$ until finding a match. With the match of (N, V), the judge and the AVP server can know the parking lots that the AV has parked and the AV will park in the future.

AV Localization. Without the user's delegation, the judge can identify the parking lot where the user's AV is parking. In specific, the judge retrieves all the parking messages $\mathcal{P}$ and tests $S \stackrel{?}{=} \hat{e}(H_g(I, N, U)^I, R^{1/t})$, until a message $\mathcal{P}^*$ succeeds the equation.

B. Property Extension

Smartphone Loss: The smartphone loss results in the troubles on vehicle pickup. Although the physical key can be used as a backup solution for AV driving, the location should be delivered to the user in advance. Thus, the *Location Query* and *AV Localization* algorithms are designed to enable the user to acquire the AV's location. Concretely, the user can query the AV's location by executing the *Location Query* algorithm when the smartphone is in hand. If the smartpone is lost or power-off, the user can use the physical key for starting the AV. If the smartphone is lost or power-off before the user acquires the location, he/she has to contact with the judge and obtain the parking lot with the assistance of the judge by performing *AV Localization* algorithm.

Vehicle Loss: When a user claims the AV loss accident, the judge firstly ensures that the user does not slander the parking lot $\mathcal{L}$ using *User Slandering Prevention*. If the claim is confirmed, the judge can find $\mathcal{L}$ for investigation and $\mathcal{L}$ cannot deny the AV loss due to *Parking Lot Hiding Prevention*. To find the lost AV, the judge can trace the AV using the *AV Tracing* algorithm with (I, U, h^w) given by the user when the attacker parks it in a parking lot. Additionally, if a stolen AV is found by the police, the judge can read the read-only memory of the OBU to acquire U and thereby learn the AV's owner.

User Slandering Prevention: To prevent a greedy user from slandering the parking lot for AV loss, the judge is capable to collect information from the AVP server to verify whether the AV loss claim is valid or not. Specifically, if the user claims the AV N is lost when it is parking in $\mathcal{L}$, the judge needs to use the *AV Tracing* algorithm to ensure that the user's AV was parking in $\mathcal{L}$. Then, the judge needs to check whether the user has picked it up or not. In doing so, the judge retrieves all $\mathcal{D}$ for AV pickup in $\mathcal{L}$ from the AVP server, performs the *User Tracing* algorithm and verifies $\mathcal{SPK}$ in each $\mathcal{D}$ to learn whether the AV has been picked up by the user. If yes, the user would be accused for slandering; otherwise, the judge confirms the AV loss in $\mathcal{L}$, and thereby starts the investigation. Of cause, the judge needs to check whether the user visited the parking lot and drove the AV away using the physical key through cameras in the parking lot.

We ensure that the attacker is unable to slander an honest user for scratching the AV since the attacker's and the user's AV can be traced using the *AV Tracing* algorithm. If either AV is not scratched, the attacker cannot succeed in claiming the scratching accident.

Hiding Prevention: If the AV is involved in accusation, it is necessary for the judge to trace the user who picked up the AV using the *AV Tracing* algorithm. Also, the user can be traced using the *User Tracing* algorithm to prevent any misbehavior. In addition, the parking lot cannot deny the parking event if the AV is proved to be lost in the parking lot. Although the parking lot is not responsible to compensate the user, it has the duty to corporate with the judge for investigation. To avoid troubles during investigation, the parking lot may hide itself from being detected that the AV was in the parking lot. Therefore, to prevent the hiding of the parking lot, when the user exposes (I, U, h^w), the judge can check

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

the signature of the parking lot (B, b, R') as $\hat{e}(B, Lh^b) \stackrel{?}{=} \hat{e}(g, R)^{-t}\hat{e}(g_0, h)\hat{e}(g_1, V)\hat{e}(g_2, h^w)\hat{e}(g_3, h^N)\hat{e}(g_4, R')$. If the signature is valid, the parking lot cannot repudiate the presence of the AV in the parking lot.

V. Security Analysis

In this section, we prove that our PrivAV achieves all the design objectives, namely, smartphone-autonomous vehicle (SP-AV) authentication, AVP authentication, AV location privacy, user anonymity and correct tracing.

SP-AV Authentication: The construction of SP-AV authentication is similar with the Waters' signature [27] in type-3 pairing. In PrivAV, the user authenticates to the AV using the one-time password pwd and the shared SP-AV key w. If both pwd and w are correct, the AV confirms that the pickup message is from the owner and starts the engine to pick up the owner at the right place and time. To keep the AV secure, it is critical to guarantee that no adversary is able to generate a valid authentication tag J, even it can acquire pwd or w.

Theorem 1. The adversary $\mathcal{A}$ cannot succeed the authentication to the AV even if $\mathcal{A}$ has the smartphone storing the SP-AV key w or guesses the one-time password pwd successfully if the CDH assumption in $\mathbb{G}_1$ holds.

Proof. The security proof consists of two parts, one proves that $\mathcal{A}$ with w cannot succeed the identity verification without pwd; and the other shows that $\mathcal{A}$ is unable to pass the authentication without w if pwd is known. The security of the former is obvious, as pwd is one-time password. Even though $\mathcal{A}$ obtains many pwd-J pairs, $\mathcal{A}$ cannot generate a valid authentication tag J^* if the new password pwd^* is unknown. Therefore, the security of authentication depends on the confidentiality of the one-time password pwd.

If $\mathcal{A}$ successfully obtains a user's pwd, the authentication can be reduced to the CDH assumption in $\mathbb{G}_1$. That is, if $\mathcal{A}$ can succeed the AV's authentication, we show how to construct a simulator $\mathcal{S}$, which can solve the CDH problem, in which given $g^x, g^y \in \mathbb{G}_1^2$, for $x, y \in_R \mathbb{Z}_p^2$, to compute g^{xy}. The view of $\mathcal{A}$ is provided by $\mathcal{S}$ with the random oracle. The security model of digital signature is utilized to formalize $\mathcal{A}$'s behaviors, as we use the signature to achieve SP-AV authentication. $\mathcal{S}$ chooses $z_0, z_1 \in_R \mathbb{Z}_p^2$, generates $Param$ and sets $W = g^x$, $g_2 = g^y$, $g_0 = (g^y)^{a_0} g^{b_0}$ and $g_1 = (g^y)^{a_1} g^{b_1}$, where a_0, a_1 are randomly chosen from $\{-1, 0, 1\}$, and b_0, b_1 are randomly chosen from $\{1, 2, \cdots, p\}$. $\mathcal{S}$ interacts with $\mathcal{A}$ in each of possible interactions.

- To respond the hash queries of $H(pwd_i||0)$ and $H(pwd_i||1)$, in which pwd_i is randomly chosen by $\mathcal{A}$, $\mathcal{S}$ randomly picks $s_i, s_i' \in_R \mathbb{Z}_p^2$, sets $s_i = H(pwd_i||0)$ and $s_i' = H(pwd_i||1)$, and returns (s_i, s_i') to $\mathcal{A}$. $\mathcal{S}$ maintains a list to keep (s_i, s_i', pwd_i).
- $\mathcal{S}$ also responds the queries of J generation. With the one-time password pwd_i, $J_i = (g_0^{H(pwd_i||0)} g_1^{H(pwd_i||1)})^{\tau_i} g_2^w = g^{xy}((g^y)^{a_0 s_i} g^{b_0 s_i} (g^y)^{a_1 s_i'} g^{b_1 s_i'})^{\tau_i} = g^{xy + \tau_i(y a_0 s_i + y a_1 s_i' + b_0 s_i + b_1 s_i')}$. If we set $g^{\tau_i} = g^{\alpha_i x + \beta_i}$, then, $J_i = g^{xy + (\alpha_i x + \beta_i)(y a_0 s_i + y a_1 s_i' + b_0 s_i + b_1 s_i')} = (g^{xy})^{1 + \alpha_i(a_0 s_i + a_1 s_i')} (g^x)^{\alpha_i(b_0 s_i + b_1 s_i')} (g^y)^{\beta_i(a_0 s_i + a_1 s_i')} + g^{\beta_i(b_0 s_i + b_1 s_i')}$. Thus, based on $a_0 s_i + a_1 s_i'$, we can define two cases:
  - If $a_0 s_i + a_1 s_i' \neq 0$, $\mathcal{S}$ can generate valid authentication tags by setting $\alpha_i = (a_0 s_i + a_1 s_i')^{-1} \mod p$ and choosing β_i randomly;
  - If $a_0 s_i + a_1 s_i' = 0$, $\mathcal{S}$ aborts and returns failure.

  Finally, suppose $\mathcal{A}$ generates a valid authentication tag J^* with a one-time password pwd^*. The results of pwd^* returned from the hash query is (s^*, s'^*).
  - If $a_0 s^* + a_1 s'^* \neq 0$, $\mathcal{S}$ aborts and returns failure;
  - If $a_0 s^* + a_1 s'^* = 0$, $\mathcal{S}$ can extract g^{xy} from $J^* = (g^{xy})(g^x)^{\alpha_i(b_0 s^* + b_1 s'^*)} + g^{\beta_i(b_0 s^* + b_1 s'^*)}$, that is, $g^{xy} = J^*((g^x)^{\alpha_i(b_0 s^* + b_1 s'^*)} + g^{\beta_i(b_0 s^* + b_1 s'^*)})^{-1}$. Thus, the CDH problem can be solved.

AVP Authentication. To prevent an adversary to maliciously access AVP for AV pickup, the AVP authentication is achieved based on the BBS+ signature. We ensure that only an accessible user who has a valid service credential can use the remote pickup service, which may be charged.

Theorem 2. The adversary $\mathcal{A}$ cannot succeed AVP authentication of an AV if the $q-$SDH assumption in $\mathbb{G}_1$ holds.

Proof. We assume that the zero-knowledge proofs $\mathcal{PK}_1$, $\mathcal{PK}_2$ and $\mathcal{SPK}$ are sound, that is, there exist extract algorithms $\mathcal{EX}_1$, $\mathcal{EX}_2$ and $\mathcal{EX}_S$ to capture the witnesses in $\mathcal{PK}_1$, $\mathcal{PK}_2$ and $\mathcal{SPK}$, respectively. The security model in [17] is utilized to formalize $\mathcal{A}$'s behaviors, as we use the signature to realize AVP authentication. These proofs can be constructed non-interactively based on Σ-protocols and are sound in the random oracle model.

The simulator $\mathcal{S}$ interacts with $\mathcal{A}$. $\mathcal{S}$ is given $Param$ and the public key β. $\mathcal{S}$ is able to access the BBS+ signature oracle $\mathcal{SO}$ that can output a BBS+ signature on a query.

- $\mathcal{A}$ randomly chooses an identity $I \in_R \mathbb{Z}_p$ and $w, u \in_R \mathbb{Z}_p^2$. $\mathcal{A}$ also computes $W = g^w$, generates a zero-knowledge proof $\mathcal{PK}_1$ and queries the registration to $\mathcal{S}$. $\mathcal{S}$ extracts (s', u, w, r') from $\mathcal{PK}_1$ using $\mathcal{EX}_1$. Then, $\mathcal{S}$ issues a signature query to $\mathcal{SO}$ and obtains (A, e, s, r). $\mathcal{S}$ computes $s'' = s - s'$ and $r'' = r - r'$. Finally, $\mathcal{S}$ returns (A, e, s'', r'') to $\mathcal{A}$.
- $\mathcal{A}$ generates a proof $\mathcal{SPK}$ and computes the pickup request $\mathcal{D}$ based on the queried service credentials. $\mathcal{S}$ returns the verification results to $\mathcal{A}$.

Finally, $\mathcal{A}$ generates a valid pickup request $\mathcal{D}^*$. From $\mathcal{SPK}$ in $\mathcal{D}^*$, $\mathcal{S}$ extracts the service credential (A^*, e^*, s''^*) that has not queried. Thus, $\mathcal{S}$ believes that $\mathcal{A}$ must conduct a valid credential (A^*, e^*, s''^*), which is a successful forge of the BBS+ signature. Since the unforgeability of the BBS+ signature is based on the $q-$SDH assumption, the security of the AVP authentication relies on the $q-$SDH assumption.

AV Location Privacy. The AV's location privacy will not be corrupted by unauthorized parties, as the AV's information is well protected in the parking message $\mathcal{P}$. In $\mathcal{P}$, although $\mathcal{L}$ is exposed to others, $\mathcal{A}$ cannot learn any knowledge about the AV and its owner. Specifically, the location information AV_L is encrypted by ψ_1, such that only the user is eligible to access it. $R = T^\gamma$ has no information about the AV. (C', R) is the ciphertext of $g_1^v g_2^w g_3^N$ based on the ElGamal

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

scheme, so the identity information is well protected due to the security of ElGamal scheme in asymmetric groups. According to the indistinguishability of the ElGamal scheme on two candicate plaintexts in asymmetric groups, it is infeasible for $\mathcal{A}$ to distinguish two possible AVs with (v_0, w_0, N_0) and (v_1, w_1, N_1) from the challenging ciphertext (R_b, C_b') of $g_1^{v_b} g_2^{w_b} g_3^{N_b}$, where $b \in \{0,1\}$. We can also view that C' is a Pedersen commitment [28] on the message (v, w, N). Since γ is randomly chosen from $\mathbb{Z}_p$, C' is random in $\mathbb{G}_1$. Besides, $\mathcal{PK}_2$ is a zero-knowledge proof, it does not disclose the witness (w, v, N, I, γ). Therefore, $\mathcal{A}$ cannot invade the AV's privacy from $(\mathcal{L}, E_w(AV_L), E_\beta(code), C', R, \mathcal{PK}_2)$ in $\mathcal{P}$. Finally, the rest work is to show that S would not expose any information about the AV.

Theorem 3. An adversary $\mathcal{A}$ cannot distinguish a specific AV from two possible honest AVs from the parking messages, if the DBDH assumption in $\mathbb{G}_T$ holds.

Proof. Our security proof reduces the AV's privacy to the DBDH problem in $\mathbb{G}_T$. That is, if $\mathcal{A}$ can distinguish the AV based on S, we can show how to construct a simulator $\mathcal{S}$ to solve the DBDH problem in $\mathbb{G}_T$, i.e., given $\hat{g}^x \in \mathbb{G}_1$, $\hat{h}^y, \hat{h}^z \in \mathbb{G}_2^2$, $\Delta = \hat{e}(\hat{g}, \hat{h})^\delta \in \mathbb{G}_T$, where $x, y, z, \delta \in_R \mathbb{Z}_p^4$, to determine whether $\hat{e}(\hat{g}, \hat{h})^{xyz} = \hat{e}(\hat{g}, \hat{h})^\delta$. $\mathcal{S}$ generates the public parameter $Param$, sets $g = \hat{g}^x$, $T = \hat{h}$ and $h = \hat{h}^y$, and sends $(Param, g, T, h)$ to $\mathcal{A}$.

Then, $\mathcal{S}$ interacts with $\mathcal{A}$ to respond the hash queries. Specifically, $\mathcal{A}$ randomly selects (I_i, N_i, U_i) to query $\mathcal{S}$. $\mathcal{S}$ randomly selects $\epsilon_i \in_R \mathbb{Z}_p$ to compute $H_i = H_g(I_i, N_i, U_i) = \hat{g}^{\epsilon_i}$ and returns it to $\mathcal{A}$. $\mathcal{S}$ also builds a list to store the tuple $(I_i, N_i, U_i, \epsilon_i, H_i)$.

Finally, $\mathcal{S}$ chooses two AVs' information (I_0, N_0, U_0) and (I_1, N_1, U_1). $\mathcal{S}$ chooses $b \in \{0,1\}$ and generates $R_b = \hat{h}^z$ and $S_b = \Delta^{I_b \epsilon_b}$, and sends them to $\mathcal{A}$. $\mathcal{A}$ guesses $b^* \in \{0,1\}$. If $b = b^*$, $\mathcal{S}$ believes that $\hat{e}(\hat{g}, \hat{h})^{xyz} = \hat{e}(\hat{g}, \hat{h})^\delta$, and this is a DBDH tuple; otherwise, this is a non-DBDH tuple.

In addition, from this security proof, it is straightforward to see that the user's identity is preserved as well against $\mathcal{A}$ in the parking message $\mathcal{P}$.

User Anonymity. The user anonymity is achieved during AV pickup against unauthorized parties, including the AVP server and the local server in the parking lot, except the AV. In the pickup request $\mathcal{D}$, the user's identity information (I, U, W) is hidden during verification against the AVP server. Concretely, $E_w(U_L, Time)$ will not expose the user's identity, as well as the location U_L, and (F, G) is the ciphertext of $G_0^s G_1^u G_2^w G_4^r$ for the judge based on the ElGamal scheme. As the ElGamal scheme is secure in group $\mathbb{G}_T$, the plaintext $G_0^s G_1^u G_2^w G_4^r$ only can be recovered by the judge, so that no adversary is able to identify the user based on (F, G). According to the indistinguishability of the ElGamal scheme in group $\mathbb{G}_T$, it is infeasible for $\mathcal{A}$ to distinguish two possible users with (I_0, U_0, W_0) and (I_1, U_1, W_1) from the challenging ciphertext (F_b, G_b) of $G_0^s G_1^u G_2^w G_4^r$, where $b \in \{0,1\}$. Besides, the zero-knowledge proof $\mathcal{SPK}$ is used to prove all relations without exposing the witness $(A, e, s, u, w, I, r, \tau)$. Therefore, the user anonymity is protected against malicious attackers and curious entities, as long as the authentication tag J would not expose users' privacy.

Theorem 4. An adversary $\mathcal{A}$ cannot distinguish a specific user from two possible honest users from pickup requests, if the DBDH assumption in $\mathbb{G}_T$ holds.

Proof. Our security proof reduces user anonymity to the DBDH problem in $\mathbb{G}_T$. That is, if $\mathcal{A}$ can distinguish the user based on J, we can show how to construct a simulator $\mathcal{S}$ to solve the DBDH problem in $\mathbb{G}_T$, i.e., given $g^x, g^y \in \mathbb{G}_1$, $\hat{h}^z \in \mathbb{G}_2$, $\Delta = \hat{e}(g, \hat{h})^\delta \in \mathbb{G}_T$, where $x, y, z, \delta \in_R \mathbb{Z}_p^4$, to determine whether $\Delta = \hat{e}(g, \hat{h})^{xyz}$. We prove even $\mathcal{A}$ can have the one-time password pwd, it cannot learn user's identity from J. $\mathcal{S}$ is allowed to access the CDH oracle that can output a CDH result given a CDH query. $\mathcal{S}$ generates the public parameter $Param$, sets $g_0, g_1 \in_R \mathbb{G}_1^2$, $W = g^x$, $g_2 = g^y$ and $h = \hat{h}^z$, and sends $(Param, W, g_0, g_1, g_2, h, \hat{h})$ to $\mathcal{A}$.

Then, $\mathcal{S}$ interacts with $\mathcal{A}$ to respond the hash queries. Suppose $\mathcal{A}$ randomly selects pwd_i to query the hash results. $\mathcal{S}$ randomly selects $\theta_i, \theta_i' \in_R \mathbb{Z}_p^2$ to respond to $\mathcal{A}$. $\mathcal{S}$ also builds a list to store the tuple $(pwd_i, \theta_i, \theta_i')$.

Finally, $\mathcal{S}$ chooses two users' information $W_0 = W g^{a_0}$ and $W_1 = W g^{a_1}$, where $a_0, a_1, \in \mathbb{Z}_p^2$. $\mathcal{S}$ chooses $b \in \{0,1\}$ and $\tau^* \in \mathbb{Z}_p$ to generate $F_b = h^{\tau^*}$. $\mathcal{S}$ can use (W_b, g_2) to query the CDH oracle and obtain $g_2^{x+a_b}$. Then, $\mathcal{S}$ computes $J_b = (g_0^{\theta^*} g_1^{\theta'^*})^{\tau^*} g_2^{x+a_b}$, and sends (F_b, J_b) to $\mathcal{A}$. $\mathcal{A}$ guesses $b^* \in \{0,1\}$. If $b = b^*$, $\mathcal{S}$ believes that $\hat{e}(g, \hat{h})^{xyz} = \hat{e}(g, \hat{h})^\delta$, and this is a DBDH tuple; otherwise, this is a non-DBDH tuple.

Correct Tracing. Correct tracing consists of two aspects, namely, slandering and hiding. In slandering, the attacker claims the AV loss or scratching accident to the judge. The judge can recover the identity information of users and AVs based on the transcripts, including registration transcripts, parking messages or pickup requests. In specific, the judge can recover the AV from a parking message $\mathcal{P}$ by using (N, V, h^w) to test $\hat{e}(C', h) = \hat{e}(g, R)^{-t} \hat{e}(g_1, V) \hat{e}(g_2, h^w) \hat{e}(g_3, h^N)$. To trace the user, the judge can link the pickup request $\mathcal{P}$ to the registration transcripts through the commitment C, and thereby learn (I, W) of the user. Through the user and AV tracing, the judge can know whether the claimed AV is really lost in the parking lot for preventing the attacker from slandering the parking lot. Also, both scratched AVs can be found by the judge for scratching investigation if a scratching accident is claimed.

Hiding: A user or AV cannot prevent itself from being traced by the judge since the user's or AV's identity should be involved in generating the pickup request $\mathcal{D}$ or the parking message $\mathcal{P}$, respectively. To prevent being traced, an attacker (malicious AV or user) has to use forged identity information and secret key to conduct $\mathcal{D}$ or $\mathcal{P}$. This is infeasible based on the unforgeability of the underlying BBS+ signature and the soundness of $\mathcal{PK}_2$ and $\mathcal{SPK}$. In addition, a malicious parking lot cannot hide itself from being traced since its signature on C' is added into the parking message, which is a BBS+ signature on (γ, v, w, N). Since the BBS+ signature is unforgeable based on the $q-$SDH assumption, the hiding of the parking lot is impossible if the $q-$SDH assumption holds.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

VI. Efficiency Analysis

We implement a prototype that simulates the computing tasks of the smartphone, AV, AVP server and local server in User Registration, AV Parking, and AV Pickup of PrivAV. In user side, we uses a smart phone, Huawei MT2-L01 with CPU Kirin 910 @1.6GHz with 1250M Memory. The computing tasks of AVP server and local server are executed on a laptop with Intel Core i5-4200U CPU @2.29GHz and 4.00GB memory. The operation system is 64-bit Windows 10 and the C++ compiler is Visual Studio 2008. To simulate AV, we change the CPU frequency of the smart phone to be 1196MHz by using a software, since the widely used Intel Atom® processor E3900 series on vehicles have 1.1–1.6 GHz CPU HFM frequency. MIRACL library 5.6.1 is used to implement cryptographic computations. p is a large prime with approximately 160 bits. We use Barreto–Naehrig curve [29], i.e., $\mathbb{F}_p$-256BN, $E : y^2 = x^3 + 3$ over $\mathbb{F}_p$. z is an integer, $n = 36z^4 + 36z^3 + 18z^2 + 6z + 1$ and $p = 36z^4+36z^3+24z^2+6z+1$ are prime and $\#E(\mathbb{F}_p) = n$. The embedding degree $k = 12$ is the smallest positive integer with $n|p^4 - 1$. $E[n] \subsetneq E(\mathbb{F}_{p^{12}})$, where $E(n)$ denotes the set of all $n-$torsion points on E. $\mathbb{G}_1 = E(\mathbb{F}_p)$, $\mathbb{G}_2$ is the trace–0 order–n subgroup of $E(n)$, and $\mathbb{G}_T$ is the order–n subgroup of $\mathbb{F}^*_{p^{12}}$. The R-ATE pairing [29] is used to achieve the type-3 bilinear pairing. The bilinear pairings $\hat{e}(g, T)$ and $\hat{e}(H_g(I, N, U)^I, h)$ can be pre-computed by the corresponding entity. In the following, we show the measured values for execution time of each entity in Table I.

To demonstrate the communication overhead, we count the number of bytes exchanged between two entities. Assume that the length of every auxiliary information (i.e., "Q", "P", $AV_L, code$) is 20 bytes. To register at the AVP server, a user sends 248-byte (C, I, W) to the AVP server, and the AVP server returns 124-byte (A, s'', e, r'') to the user. In AV Parking phase, an AV sends the parking message $\mathcal{P}$ to the local server in the parking lot, which is 352 bytes, and the local server forwards $(\mathcal{P}, B, b, R')$ to the AVP server, which is 500 bytes. To query the location of an AV, the user sends 104-byte ("Q", $E_\beta(code), \psi_1$) to the AVP server and receives 20-byte $(E_w(AV_L), \psi_2)$. In AV Pickup phase, the smartphone generates the pickup request $\mathcal{D}$, whose binary length is 620 bytes, and sends it to the AVP server. The AVP server forwards ("P", $code', E_w(U_L, Time), J, \psi$) to the local server, which is 188 bytes. Finally, the local server broadcasts 84-byte $(code', \psi)$ to find the user's AV and 84-byte $(E_w(U_L, Time), J)$ to the AV for pickup.

We compare our PrivAV with the existing schemes, i.e., LI [13], HLKW [14] and JLYM [16], about the computational and communication overhead of the two-factor authentication. The authentication message in PrivAV is $(F, J) = (h^\tau, (g_0^{H(pwd||0)} g_1^{H(pwd||1)})^\tau g_2^w)$. To improve the efficiency, the exponentiations $g_0^{H(pwd||0)} g_1^{H(pwd||1)}$ and g_2^w can be executed after (pwd, w) are determined. Thus, only two exponentiations are executed for the smartphone in AV Pickup. The comparison results with the existing schemes are shown in Fig. 3(a). The authentication message generation in PrivAV is more efficient than that in LI [13], HLKW [14] and JLYM [16]. In the verification of (F, J), the computational overhead can be reduced by using pre-computation, since g_2^{-w} and $g_0^{H(pwd||0)} g_1^{H(pwd||1)}$ can be calculated for the AV in advance. Further, the bilinear pairings $\hat{e}(Jg_2^{-w}, h)$ and $\hat{e}(g_0^{H(pwd||0)} g_1^{H(pwd||1)}, F)$ are able to be performed with the aid of the local server. The approach is as follows: The AV chooses random values $s_1, s_2 \in_R Z_p$ to compute $S_1 = (Jg_2^{-w})^{s_1}$ and $S_2 = (g_0^{H(pwd||0)} g_1^{H(pwd||1)})^{s_2}$, and sends (S_1, S_2) to the local server; the local server calculates $S'_1 = \hat{e}(S_1, h)$ and $S'_2 = \hat{e}(S_2, F)$ and returns (S'_1, S'_2) to the AV; the AV verifies whether $(S'_1)^{s_1^{-1}} \stackrel{?}{=} (S'_2)^{s_2^{-1}}$ to determine the validity of authentication. In this way, the burden on calculating the bilinear pairings is migrated to the local server, and the local server cannot corrupt the authentication result. We compare the computational overhead on the authentication verification between PrivAV with server-aided verification and LI [13], HLKW [14] and JLYM [16], and demonstrate that the PrivAV is the most efficient scheme in four. In terms of the communication burden, Fig. 3(c) illustrates that the proposed PrivAV consumes the least communication resources in PrivAV, LI [13], HLKW [14] and JLYM [16].

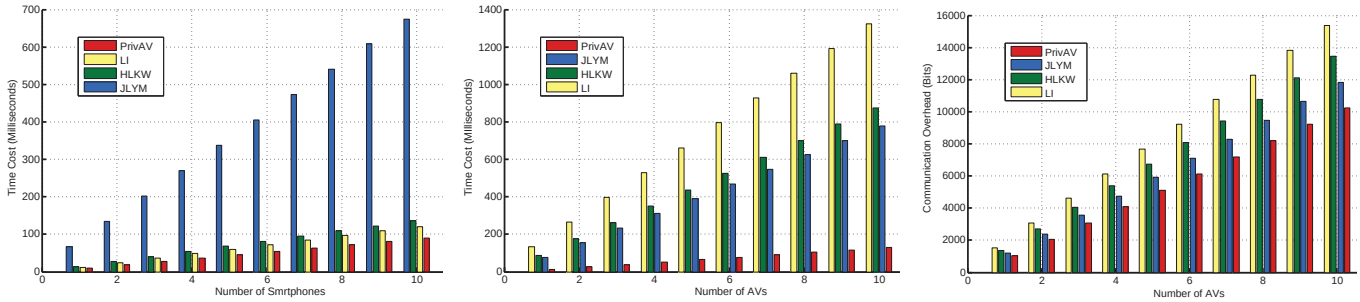
In addition, we demonstrate that the proposed PrivAV is efficient even when the density of vehicles is large. For example, numerous AVs are pending to park (or pickup) before (or after) large social events, such as football games, national day events, and special festivals. When the number of vehicles pending to park is large, the local server and the AVP server would receive numerous parking messages or pickup requests, which possess a large amount of bandwidth for data transmission and computing resources of both servers. As the result, the service delay increases. Fig. 4 shows the time costs of the local server and the AVP server on dealing with all the requests from users. As shown in Fig. 4(a), if 1000 AVs send 1000 parking messages to both servers at the same time, the local server needs to use around 27.5s, and the AVP server uses 19s to respond these requests, respectively. Fig. 4(b) illustrates that the AVP server can respond 1000 location queries from mobilephones in 14s. In AV Pickup, the local server is able to respond 1000 requests in about 5s and the AVP server only needs 11s to deal with 1000 pickup requests. Therefore, even if the number of received parking messages and the quantity of pickup requests are large, the local server and the AVP server can respond these requests rapidly. Besides, the communication burden caused by PrivAV is linear with the number of AVs submitting requests. That is, if n AVs are parked in the parking lots, $352n$-byte messages are sent to the local server, and the local server forwards $500n$-byte data to the AVP server in AV Parking. Here we only consider the size of data body, since the package header has constant size in a network package. To avoid network congestion, it is possible to deploy drone-cells to migrate the network traffic for the crowded areas.

Finally, we discuss the storage overhead of all entities. These entities are required to maintain the system parameter $Param$ that is around 1.5KB. To access the AVP service, the smartphone has to store $(I, u, U, N, V, w, W, A, e, s, r, s', r', pwd, code, L, T, \beta)$, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2894720, IEEE Transactions on Vehicular Technology

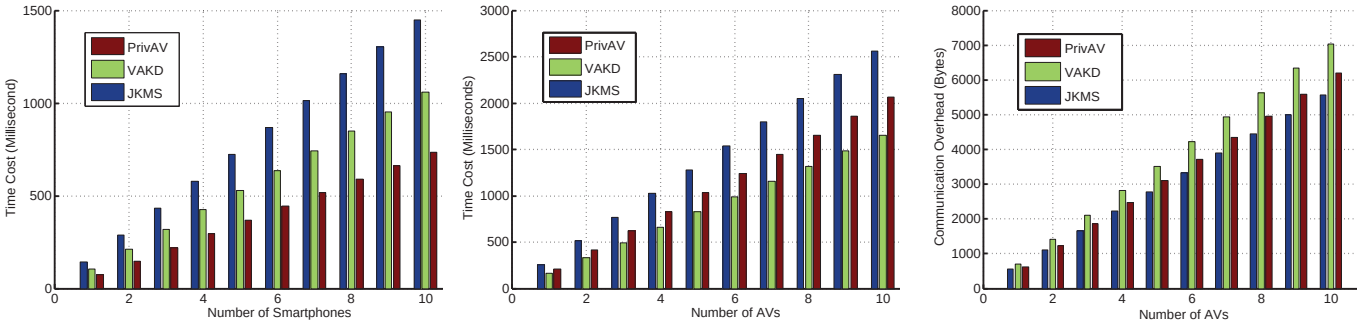
TABLE I
IMPLEMENTED RUNNING TIME (UNIT: MILLISECOND)

Phase	User Registration		AV Parking			Location Query		AV Pickup			
	Smartphone	AVP Server	AV	Local Server	AVP Server	Smartphone	AVP Server	Smartphone	AV	Local Server	AVP Server
Runtime	182.605	30.136	81.287	27.257	18.632	153.362	13.114	73.624	206.292	10.922	4.752



(a) Time Cost for Smartphones (b) Time Cost for AVs (c) Communication Overhead for Authentication

Fig. 3. Performance Comparison on Two-factor Authentication



(a) Time Costs in AV Parking (b) Time Costs in Location Query (c) Time Costs in AV Pickup

Fig. 4. Scalability of PrivAV

size of which is about 628 bytes, and the AV needs to maintain 504-byte $(v, V, w, W, U, T, L, pwd, code, I, N, \beta)$. The local server keeps the key information (T, L, l, β), the parking requests $\mathcal{P}$, and the pickup messages $(\text{“}P\text{”}, code', E_w(U_L, Time), J, \psi)$. The total size of these is 212+523n+188m bytes, where n is the number of parking requests and m is the number of pickup requests. The AVP server needs to maintain the key information (T, L, α, β), parking messages $(\mathcal{P}, B, b, R')$, and pickup requests $\mathcal{D}$, the total size of which is 212+500n+620m bytes. In addition, the AVP server also maintains the Cuckoo filter.

VII. RELATED WORK

To protect the communications between connected vehicles, considerable research efforts have been invested in identifying security vulnerabilities, recommending potential protection techniques and developing corresponding security protocols. These research works have secured a broad variety of vehicular applications [30], [31], one of which is privacy-preserving smart parking. Yan et al. [32] proposd SmartParking, a service-oriented intelligent parking system to enable drivers to securely view and reserve parking spots. Biswas and Mišić [33] presented a privacy-preserving vehicle parking assistance system based on priority-based secure vehicle-to-infrastructure communications. Lu et al. [34], [35] designed a secure smart parking scheme supporting anti-theft protection for large parking lots through the communications between vehicles and roadside units in the parking lots. However, Lu et al.'s scheme only covers a small scale. Ni et al. [36], [37] proposed a privacy-preserving smart parking navigation system for vehicles on roads supporting privacy leakage prevention for drivers and navigation results retrieval for improving the probability of successful navigation query. Although the drivers' privacy can be protected during parking assistance, their identities may still be exposed during parking fee payment in the above schemes. To address this issue, Garra [38] developed a mobile payment system via phones based on anonymous electronic coin. However, in the coming autonomous driving era, privacy preservation of users, which is the design goal of the above schemes, is insufficient to free users from the security concerns in AVP services.

Due to the wide network connectivity, autonomous vehicles are confronted with a heightened risk of cyber security attacks, which raise extensive concerns to drivers on autonomous driving. The potential cyberattacks on AVs have been investigated [39] with their special needs and vulnerabilities. A technical and social analysis [40] was proposed based on a methodological approach to encourage the future secure

and privacy-preserving solutions. The demands on protecting data integrity, driving safety and privacy preservation in the emerging autonomous driving scenarios were discussed [41] and an exemplary case of map updates was also studied to exemplify the privacy-enhanced techniques and integrity-protecting mechanisms. To preserve location privacy, Joy and Gerla [42] described several vehicular applications and proposed the concept of haystack privacy to strengthen privacy and maintain accuracy. To preserve the privacy in AV sharing service, Hadian et al. [43] proposed a privacy-preserving time-sharing scheme enabling the AV owner and requester to negotiate time schedules for AV usage with privacy preservation. Consequently, Sherif et al. [44] introduced ride sharing services for AVs and presented a similarity measurement mechanism over encrypted travel data to achieve privacy-preserving ride arrangement for service providers. On securing interactions between AVs, Amoozadeh et al. [45] identified the risks and requirements on the security of the communication channels in cooperative adaptive connected vehicle streams. To clarify the threats of cyberattacks and the current research efforts on security protection in autonomous driving, Parkinson et al. [8] identified the typical vulnerabilities in on-board control systems and communication networks, and review the research on AV related cybersecuirty mitigation techniques. They also revisited the existing research outcomes and human-centric security design methods to reduce the likelihood of successful cyberattacks on AVs.

Although the security weaknesses and requirements of AVs have been clarified, the study of security protection is still in its infancy. Many essential AV services have not received their deserved attentions. In this paper, we proposed a privacy-preserving automated valet parking service for AVs to achieve secure authentication for vehicle remote pickup, user privacy preservation for AVP service access, and the traceability of anonymous users for slandering prevention and fairness maintenance.

VIII. Conclusions

In this paper, we have proposed a privacy-preserving automated valet parking protocol (PrivAV) for self-driving vehicles. PrivAV achieves two-factor authentication based on one-time password and secure smartphone to guarantee AV security in remote retrieval, and protect user location privacy during AVP service access. In addition, to prevent slandering and maintain fairness, PrivAV enables the judge to be engaged in the AVP service for tracing anonymous users and localizing stolen AVs. With PrivAV, users can feel free to participate in the AVP services without any concern on the safety of their vehicles and the corruption of their privacy. In the future work, we will enhance the security protection on AV's control systems and design effective intrusion detection mechanisms to protect autonomous vehicles against malicious cyber attacks.

References

[1] A. Davies, "The wired guide to self-driving cars," *Wired*, Online available: https://www.wired.com/story/guide-self-driving-cars/, March 2018.

[2] L. Hobert, A. Festag, I. Llatser, L. Altomare, F. Visintainer, and A. Kovacs, "Enhancements of V2X communication in support of cooperative autonomous driving," *IEEE Commun. Mag.*, vol. 53, no. 12, pp. 64–70, 2015.

[3] H. Banzhaf, D. Nienhüser, S. Knoop, and J. M. Zöllner, "The future of parking: A survey on automated valet parking with an outlook on high density parking," *Proc. IV'17*, Redondo Beach, CA, USA, June 11–14, 2017, pp. 1827–1834.

[4] Daimler, "Driverless in the parking lot. Automated Valet Parking," Online Available: https://www.daimler.com/innovation/case/autonomous/driverless-parking.html, Jul. 2017.

[5] E. Biba, "What the world will look lile without drivers," *Newsweek*, Online Available: http://www.newsweek.com/2016/01/22/driverless-cars-and-future-getting-around-415405.html, Jan. 2016.

[6] F. Michel, et al., "Device and method for self-automated parking lot for autonomous vehicles based on vehicular networking," *Google Patents*, Online available: https://patents.google.com/patent/WO2015114592A1/en, Jan. 2015.

[7] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. USENIX Security'11*, San Francisco, CA, US, Aug. 8–12, 2017, pp. 1–16.

[8] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: future challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2898–2915, 2017.

[9] A. Drozhzhin, "Black Hat USA 2015: The full story of how that Jeep was hacked," online available: https://www.kaspersky.com/blog/blackhat-jeep-cherokee-hack-explained/9493/, August 2015.

[10] A. Francillon, B. Danev, and S. Capkun, "Relay attacks on passive keyless entry and start systems in modern cars," *Proc. NDSS'11*, San Diego, California, US, Feb. 6–9, 2011, pp. 1–15.

[11] F. D. Garcia, D. Oswald, T. Kasper, and P. Pavlidès, "Lock tt and still lose ot-on the (in)security of automotive remote keyless entry systems," *Proc. USENIX Security'16*, Austin, TX, US, Aug. 10–12, 2016, pp. 929–944.

[12] Q. Jiang, N. Kumar, J. Ma, J. Shen, D. He, and N. Chilamkurti, "A privacy-aware two-factor authentication protocol based on elliptic curve cryptography for wireless sensor networks," *Int. J. Netw. Manag.*, vol. 27, no. 3, article no. e1937, 2017.

[13] C. Li, "A new password authentication and user anonymity scheme based on elliptic curve cryptography and smart card," *IET Information Security,* vol. 7, no. 1, pp. 3–10, 2013.

[14] D. He, S. Zeadally, N. Kumar, and W. Wu, "Efficient and anonymous mobile user authentication protocol using self-certified public key cryptography for multi-server architectures," *IEEE Transactions on Information Forensics and Security,* vol. 11, no. 9, pp. 2052–2064, 2016.

[15] P. Vijayakumar, M. Azees, A. Kannan, and L. J. Deborah, "Dual authentication and key management techniques for secure data transmission in vehicular ad hoc networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1015–1028, 2016.

[16] Q. Jiang, X. Lian, C. Yang, J. Ma, Y. Tian, and Y. Yang, "A bilinear pairing based anonymous authentication scheme in wireless body area networks for mHealth," *Journal of medical systems*, vol. 40, no. 11, article. 231, 2016.

[17] M. H. Au, W. Susilo, and Y. Mu, "Constant-size dynamic k-TAA," *Proc. SCN'06*, Maiori, Italy, Sept. 6–8, 2006, pp. 111–125.

[18] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher, "Cuckoo filter: Practically better than bloom," *Proc. CoNEXT'14*, Sydney, Australia, Dec. 2–5, 2014, pp. 75–88.

[19] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for Internet of Things applications: Challenges and solutions," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 1, pp. 601–628, 2018.

[20] D. Pointcheval and O. Sanders, "Short randomizable signatures," *Proc. CT-RSA'16*, San Francisco, CA, USA, Feb. 29–Mar. 4, 2016, pp. 111–126.

[21] F. Vercauteren, et al., "Final report on main computational assumptions in cryptography," *ECRPYT II*, technical report, Jan. 2013.

[22] D. Boneh and X. Boyen, "Short signatures without random oracles and the SDH assumption in bilinear groups," *J. Cryptology*, vol. 21, no. 2, pp. 149–177, 2008.

[23] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," *Proc. CRYPTO'04*, Santa Barbara, California, USA, Aug. 15–19, 2004, pp. 41–55.

[24] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," *Proc. CRYPTO'04*, Santa Barbara, California, USA, Aug. 15–19, 2004, pp. 56–72.

[25] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM J. Comput.*, vol. 18, no. 1, pp. 186–208, 1989.

[26] W. Diffie, P. C. Oorschot, and M. J. Wiener, "Authentication and authenticated key exchanges," *Designs Codes Cryptogr.*, vol. 2, no. 2, pp. 107–125, 1992.

[27] B. Waters, "Efficient identity-based encryption without random oracles," in *Proc. EUROCRYPT'05*, Aarhus, Demmark, May 22-26, 2005, pp. 114–127.

[28] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," *Proc. CRYPTO'91*, Santa Barbara, California, USA, Aug. 11–15, 1991, pp. 129–140.

[29] J. Fan, F. Vercauteren, and I. Verbauwhede, "Faster $\mathbb{F}_p$–arithmetic for cryptographic pairings on Barreto-Naehrig curves", *Proc. CHES'09*, Lausanne, Switzerland, Sept. 6–9, 2009, pp. 240–253.

[30] N. Cheng, F. Lyu, J. Chen, W. Xu, H. Zhou, S. Zhang, and X. Shen, "Big data driven vehicular networks," *arXiv preprint arXiv:1804.04203*, 2018.

[31] Z. Su, Y. Hui, Q. Xu, T. Yang, J. Liu, and Y. Jia, "An edge caching scheme to distribute content in vehicular networks," *IEEE Trans. Veh. Technol.*, DOI: 10.1109/TVT.2018.2824345, 2018.

[32] G. Yan, W. Yang, D. B. Rawat, and S. Olariu, "SmartParking: A secure and intelligent parking system," *IEEE Intell. Transp. Syst. Mag.*, vol. 3, no. 1, pp. 18–30, 2011.

[33] S. Biswas and J. V. Misic, "Prioritized WAVE-based parking assistance with security and user anonymity," *J. Commun.*, vol. 7, no. 8, pp. 577–586, 2012.

[34] R. Lu, X. Lin, H. Zhu, and X. Shen, "SPARK: a new VANET-based smart parking scheme for large parking lots," *Proc. IEEE INFOCOM'09*, Rio De Janeiro, Brazil, April. 19–25, 2009, pp. 1413–1421.

[35] 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. 2010.

[36] J. Ni, K. Zhang, X. Lin, Y. Yu, and X. Shen, "Cloud-based privacy-preserving parking navigation through vehicular communications," *Proc. EAI SECURECOMM'16*, Guangzhou, China, Oct. 10–12, 2016, pp. 85–103.

[37] J. Ni, K. Zhang, Y. Yu, X. Lin, and X. Shen, "Privacy-preserving smart parking navigation supporting efficient driving guidance retrieval," *IEEE Trans. Veh. Technol.*, DOI: 10.1109/TVT.2018.2805759, 2018.

[38] R. Garra, S. Martínez, and F. Sebé, "A privacy-preserving pay-by-phone parking system," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 5697–5706, 2017.

[39] J. Petit and S. E. Shladover, "Potential cyberattacks on automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 546–556, 2015.

[40] Z. Ma, W. Seböck, B. Pospisil, C. Schmittner, and T. Gruber, "Security and privacy in the automotive domain: A technical and social analysis," *Proc. SAFECOM'17 Workshops*, Rento, Italy, Sept. 12, 2017, pp. 427–434.

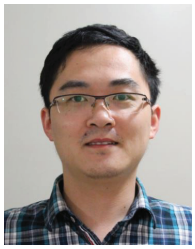
[41] S. Karnouskos and F. Kerschbaum, "Privacy and integrity considerations in hyperconnected autonomous vehicles," *Proc. IEEE*, vol. 106, no. 1, pp. 160–170, 2018.

[42] J. Joy and M. Gerla, "Internet of vehicles and autonomous connected car-privacy and security issues," *Proc. ICCCN'17*, Vancouver, Canada, Jul. 31–Aug. 3, 2017, pp. 1–9.

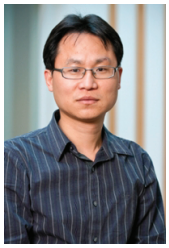
[43] M. Hadian, T. Altuwaiyan, and X. Liang, "Privacy-preserving time-sharing services for autonomous vehicles," *Proc. VTC'17-Fall*, Toronto, Canada, Sept. 24–27, 2017, pp. 1–5.

[44] A. Sherif, K. Rabieh, M. E. A. Mahmoud, and X. Liang, "Privacy-preserving ride sharing scheme for autonomous vehicles in big data era," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 611–618, 2017.

[45] M. Amoozadeh, A. Raghuramu, C. -N. Chuah, D. Ghosal, H. M. Zhang, J. Rowe, and K. Levitt, "Security vulnerabilities of connected vehicle streams and their impact on cooperative driving," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 126–132, 2015.



Jianbing Ni (M'18) received the Ph.D. degree in Electrical and Computer Engineering from University of Waterloo, Waterloo, Canada, in 2018, and received the B.E. degree and the M.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2014, respectively. He is currently a postdoctoral research fellow at the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research interests are applied cryptography and network security, with current focus on cloud computing, smart grid, mobile crowdsensing and Internet of Things.



Xiaodong Lin (M'09-SM'12-F'17) received the PhD degree in Information Engineering from Beijing University of Posts and Telecommunications, China, and the PhD degree (with Outstanding Achievement in Graduate Studies Award) in Electrical and Computer Engineering from the University of Waterloo, Canada. He is currently an associate professor in the School of Computer Science at the University of Guelph, Canada. His research interests include computer and network security, privacy protection, applied cryptography, computer forensics, and software security. He is a Fellow of the IEEE.

Xuemin (Sherman) Shen (M'97-SM'02-F'09) received Ph.D. degree from Rutgers University, New Jersey (USA) in electrical engineering, 1990. Dr. Shen is a University Professor, Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on resource management in interconnected wireless/wired networks, wireless network security, social networks, smart grid, and vehicular ad hoc and sensor networks. Dr. Shen is a registered Professional Engineer of Ontario, Canada, an IEEE Fellow, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, and a Distinguished Lecturer of IEEE Vehicular Technology Society and Communications Society.

Dr. Shen is the Editor-in-Chief for IEEE Internet of Thing Journal and the vice president on publications of IEEE Communications Society. He received the Joseph LoCicero Award in 2015, the Education Award in 2017, the Harold Sobol Award in 2018, and the James Evans Avant Garde Award in 2018 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006, and the Outstanding Performance Award in 2004, 2007, 2010, 2014, and 2018 from the University of Waterloo, the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. Dr. Shen served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom' 16, IEEE Infocom'14, IEEE VTC'10 Fall, the Symposia Chair for IEEE ICC'10, the Tutorial Chair for IEEE VTC'11 Spring, the Chair for IEEE Communications Society Technical Committee on Wireless Communications, and P2P Communications and Networking.