Anh Pham*, Italo Dacosta, Bastien Jacot-Guillarmod,
Kévin Huguenin, Taha Hajar, Florian Tramèr, Virgil Gligor, and Jean-Pierre Hubaux

PrivateRide: A Privacy-Enhanced Ride-Hailing Service

Abstract: In the past few years, we have witnessed a rise in the popularity of ride-hailing services (RHSs), an online marketplace that enables accredited drivers to use their own cars to drive ride-hailing users. Unlike other transportation services, RHSs raise significant privacy concerns, as providers are able to track the precise mobility patterns of millions of riders worldwide. We present the first survey and analysis of the privacy threats in RHSs. Our analysis exposes high-risk privacy threats that do not occur in conventional taxi services. Therefore, we propose PrivateRide, a privacy-enhancing and practical solution that offers anonymity and location privacy for riders, and protects drivers' information from harvesting attacks. PrivateRide lowers the high-risk privacy threats in RHSs to a level that is at least as low as that of many taxi services. Using real data-sets from Uber and taxi rides, we show that PrivateRide significantly enhances riders' privacy, while preserving tangible accuracy in ride matching and fare calculation, with only negligible effects on convenience. Moreover, by using our Android implementation for experimental evaluations, we show that PrivateRide's overhead during ride setup is negligible. In short, we enable privacy-conscious riders to achieve levels of privacy that are not possible in current RHSs and even in some conventional taxi services, thereby offering a potential business differentiator.

Keywords: ride-hailing, location privacy

DOI 10.1515/popets-2017-0013
Received 2016-08-31; revised 2016-09-30; accepted 2016-10-01.

***Corresponding Author: Anh Pham:** EPFL, Lausanne, Switzerland, E-mail: thivananh.pham@epfl.ch
Italo Dacosta: EPFL, Lausanne, Switzerland, E-mail: italo.dacosta@epfl.ch
Bastien Jacot-Guillarmod: Google, E-mail: bjacotg@gmail.com. This work was carried out while the author was with EPFL, Lausanne, Switzerland.
Kévin Huguenin: University of Lausanne, Lausanne, Switzerland, E-mail: kevin.huguenin@unil.ch
Taha Hajar: EPFL, Lausanne, Switzerland, E-mail: taha.hajar@epfl.ch
Florian Tramèr: Stanford University, Stanford, USA, E-mail: tramer@stanford.edu. This work was carried out while the author was with EPFL, Lausanne, Switzerland.
Virgil Gligor: CMU, Pittsburgh, USA , E-mail: gligor@cmu.edu
Jean-Pierre Hubaux: EPFL, Lausanne, Switzerland, E-mail: jean-pierre.hubaux@epfl.ch

1 Introduction

Over the last few years, the popularity of RHSs, such as Uber and Lyft, has significantly increased, serving millions of users in hundreds of cities [1]. These services provide an efficient marketplace where users can register themselves as drivers (to provide rides) and/or riders (to hail rides); the service provider (SP) matches ride requests with available drivers. More importantly, by relying on smartphones and mobile apps, RHSs bring substantial improvements to the ride-hailing experience by facilitating the requests, the paying for and the rating of the rides.

Unfortunately, along with the high convenience of RHSs come some important privacy concerns. In contrast with traditional taxi services, SPs in RHSs collect the details of each ride, including location traces, together with the rider's real identity. Note that other forms of transportation, such as taxis and private cars, also leak private information. They, however, require the SPs to do extra and error-prone operations (e.g., image processing for face/licence plate recognition) to identify users and their activities. Therefore, data collection in RHSs is more efficient and large-scale. As a result, the SP, or any third-party with access to this data, can infer privacy-sensitive information such as where riders live, work, and socialize. Unsurprisingly, a number of unethical uses of this data have already been reported: analysis of user trajectories to discover patterns of one-night stands [49], monitoring of the location of riders in real-time for entertainment [27], and even revenge attacks against journalists critical of such services [64]. This problem is exacerbated by the fact that SPs often share or sell this data to third-parties [29]. In addition, drivers' privacy is also at risk. As our privacy analysis reveals (Section 3), current RHSs do little to prevent adversaries from massively harvesting information about the drivers. Such information could be used by traditional taxi drivers to coordinate physical attacks [23].

In this paper, we analyze the privacy threats in RHSs and propose a privacy-enhanced RHS, PrivateRide; it protects riders' anonymity and location privacy, while maintaining most of the benefits of current RHSs, i.e., accountability, automatic payments, and reputation ratings. We perform the first privacy analysis of RHSs, develop a threat

taxonomy, and focus on two critical threats: location tracking of riders by the SP and harvesting of drivers' personal information by an external adversary. To target these threats, PrivateRide provides riders with anonymity and unlinkability with respect to the SP and drivers.

PrivateRide relies on well-known cryptographic and privacy-enhancing building blocks (e.g., anonymous credentials, blind signatures, e-cash, location and time cloaking). These building blocks are carefully composed to achieve our privacy and security goals, as well as to support RHSs' key services: ride matching, fare calculation and estimation, reputation rating for drivers and riders, payments, and accountability. Note that our contribution is not the design of new techniques, rather the selection, composition, and evaluation of well-known, simple, and efficient techniques. Such an approach is crucial for deployment in real systems because these techniques are easier to understand and accept by users and developers.

To evaluate PrivateRide's privacy gains and effects on usability, we use real data-sets from NYC taxi rides [63] and SF Uber [32]. We show that, at peak hours in Manhattan, our location and time cloaking techniques can provide an anonymity set of 4 and 10 for 80% and 40% of the rides, respectively (Section 6); thus, hindering strong targeted attacks where the adversary knows the approximate time and location a particular rider hailed a ride. Moreover, it is easy to see that cloaking offers a robust defense against large-scale inference attacks, where the adversary has much less information about the riders (e.g., analysis of one-night stands [49]). Note that PrivateRide offers by default a lower-bound anonymity set to all riders. Privacy-conscious riders can take actions to improve their privacy, e.g., by walking to a more crowded cloaked area to hail a ride. Furthermore, we demonstrate that PrivateRide has a negligible effect on usability. We observe that 95% of the rides have less than 10% error on fare calculation, and 80% of the rides introduce less than 100 m of overhead on the total pick-up distance (Section 8). In addition, by using PrivateRide's Android implementation, we show that the added overhead is negligible, i.e., just a few seconds (most ride-hailing operation take minutes [28]).

In short, our main contributions are:

- We present the first general privacy analysis of RHSs. By analyzing currently deployed RHSs, we develop a threat taxonomy and identify high-risk threats, particularly the unreported threat of drivers' personal information being harvested.
- We propose PrivateRide, the first practical system that offers enhanced privacy for riders and drivers, without affecting the convenience of these services or the SP's economic incentives. To facilitate adoption, we rely exclusively on well-established, efficient cryptographic and privacy-enhancing primitives.
- We analyze and evaluate, by using real data-sets, PrivateRide's privacy guarantees and effect on usability. We show that PrivateRide offers significant privacy gains for all riders and a means for privacy-conscious riders to further enhance their privacy. Moreover, we demonstrate that PrivateRide introduces a negligible performance overhead by using our Android implementation.

Current RHSs

Rider's **real identity**	**Precise** pickup and drop-off locations	**Precise** pickup and drop-off times	**Full** location trace	Fare	Driver's real identity

PrivateRide

Rider's **anonymous identity**	**Cloaked** pickup and drop-off locations	**Cloaked** pickup and drop-off times	**Partial** location trace	Fare	Driver's real identity

Fig. 1. Information collected by the SP per ride for current RHSs and PrivateRide. PrivateRide cloaks rides' sensitive information to protect against reidentification attacks.

Figure 1 summarizes the differences between the information that an SP gathers per ride in current RHSs and PrivateRide. By cloaking sensitive information, PrivateRide makes inference attacks more difficult, i.e., an adversary requires more side information and the SP can still benefit from the cloaked information.

2 Ride-Hailing Services

In general, RHSs involve three parties: riders, drivers, and a service provider (SP) (see Figure 2). The SP handles incoming ride-requests and matches riders with available drivers, based primarily on their locations; it also offers key services such as fare estimation and calculation (based on the route of the rides), ride payment and reputation management. In exchange for these services, the SP charges a fee for each completed ride (e.g., Uber charges around 20% of the total fare). Some SPs also sell ride data or traces to third-parties (e.g., for city planning [15] or marketing [29]). To use a RHS, riders and drivers need an account, a GPS-equipped smartphone with the SP's mobile app installed, and an active Internet connection.

To hail a ride, the rider sends a request to the SP by using the mobile app. The request includes the rider's identity and the exact pick-up and (optionally) drop-off locations. The SP selects an available driver, based on her

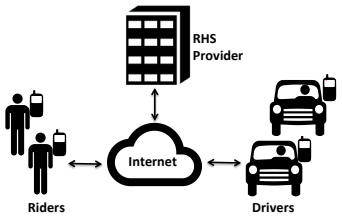


Fig. 2. Ride-hailing services overview.

Description			Risk-RH	Risk-T
1) R→	D	PII harvesting [2]	Low	Low
2) D→	R	PII harvesting	Low	Low
3) **SP→**	**R**	**location tracking [27, 31, 49]**	**High**	Medium
4) SP→	D	location tracking [47]	Medium	Low
5) O→	R	PII harvesting	Medium	Low
6) **O→**	**D**	**PII harvesting**	**High**	Low
7) **O→**	**SP**	**PII and ride data breach [3]**	**High**	Medium

Table 1. Privacy threat taxonomy for RHSs (Risk-RH) and traditional taxi services (Risk-T) based on four possible adversaries: riders (R), drivers (D), service provider (SP), and outsiders (O). The notation $X \rightarrow Y$ means *X attacks Y*.

proximity to the pick-up location (drivers continuously report their locations to the SP while on duty). The SP sends the request, together with the rider's username, reputation and phone number (for ride coordination) to the driver. If the driver declines the request, the SP sends the request to another available driver. Otherwise, the SP notifies the rider and sends her back the driver's information such as the driver's real name, photo, phone number, and car plate number. The rider uses this information to decide if she accepts the ride, as well as to coordinate the pickup.

As the driver approaches the pick-up location, the SP shares the driver's location and estimated time of arrival with the rider. Once the rider is in the car, the driver notifies the SP that the ride has begun. Both the rider and the driver can cancel a ride at any point. However, in certain cases, the SP can charge them a penalty fee or lower their reputations [58] (e.g., in case of systematic ride cancellation). Unlike the rider, the driver must keep her smartphone switched on (with Internet and GPS connectivity) during the ride, in order to report her location to the SP. Upon reaching the destination, the driver notifies the SP that the ride has ended. The SP computes the fare, based on the location information reported by the driver, and automatically charges the rider (e.g., her credit card); the amount charged to the rider (minus the service fees) is credited to the driver's account. Both the driver and rider can check the details of the ride via the mobile app and rate each other. This operation is optional and it can be performed within a predefined period of time that begins right after the end of the ride. The SP uses the reputation information to keep the quality of the service high by detecting and punishing misbehaving parties. For example, riders and drivers with low reputations could be banned temporarily or permanently.

3 Privacy Analysis of RHSs

RHSs introduce significant privacy threats to riders and drivers. In this section, we present a privacy analysis of current RHSs and identify the most critical privacy threats in these services. Appendix A presents an analysis of integrity threats in RHSs.

3.1 Adversarial Model

We define four adversaries in current RHSs:

Rider. Active adversary who might attempt to harvest personally identifiable information (PII) from drivers (e.g., for stalking or blackmailing purposes).

Driver. Active adversary who might attempt to collect PII from riders she picks up (e.g., for stalking or blackmailing purposes).

SP. Passive adversary that strives to safeguard its business and maximize its profits. It has incentives to profile riders and drivers and infer sensitive information about them, to either improve its own services or to monetize harvested data (e.g., for advertisement purposes or coercion [64]). However, it also has incentives to prevent certain attacks, such as data harvesting from outsiders, or pervasive fare overcharging, as these attacks threaten the viability of the SP's business over its competitors. Finally, we assume the SP has no incentive to actively *cheat* (e.g., by providing a malicious app to users, or otherwise deviating from an established protocol), if there is a non-negligible chance of the SP being caught in the act. Indeed, the risk of public exposure and reputation loss is a strong economic deterrent against such attacks from the SP.

Outsider. Active adversary who tries to collect and/or steal riders' and drivers' PII, account credentials, and ride data. The malicious outsider is not part of the RHS and might have more resources than a single driver or rider. For instance, it could be a competitor SP, a criminal organization, or a group of regular taxi drivers.

3.2 Threat Taxonomy

We built the first threat taxonomy for privacy threats in RHSs (Table 1). To identify these threats, we reviewed online sources (e.g., news articles, RHSs websites, forums, blog posts) and experimentally evaluated some of the most popular RHSs, see Section 3.3 for more details. To estimate the level of risk associated with each adversary and threat, we followed the OWASP risk-rating methodology [50] where the risk is defined as: $Risk = Impact * Likelihood$. In our risk assessment, we assume that only a small subset of the riders and drivers are malicious. This is a reasonable assumption, given the current success of RHSs.

In general, threats with low risks involve attacks that are not scalable, offer limited rewards, and can be deterred by existing mechanisms such as decreased reputation [58, 59]. Threats with medium risks involve attacks that have relatively higher impact, offer higher rewards, and are more likely to happen. Threats with high risks involve attacks that have the highest impact and reward, have the highest likelihood and reported incidents, and for which current defense mechanisms are insufficient. Appendix B details each threat and justifies each risk level.

In Table 1, we also show the risk levels of each privacy threat for traditional taxi services (i.e., Risk-T). Risk levels in taxi services are lower than in RHSs (i.e., Risk-RH), because SPs in the former collect less sensitive information. The only exceptions are taxi services where the SP collect riders' PII and ride details for reservations or business analytics purposes. Hence, threats 3 and 7 in Table 1 are rated as medium risk. In this paper, our goal is to *lower the high-risk privacy threats in RHSs to a level that is at least as low as that of many taxi services.*

From the high-risk threats in RHSs in Table 1, threat 7 is not exclusive to RHSs; data breaches affect almost any online service. Hence, there are existing mechanisms to reduce its risk. For example, the SP database could use CryptDB [53] to securely store riders' and drivers' data.[1] Therefore, in this work we focus on the high-risk threats that *occur exclusively due to the design of current RHSs and for which there are no current solutions* (i.e., threats 3 and 6 on Table 1):

SP→R location tracking. By design, an SP can track, in real time or offline, riders' precise locations during their rides and infer private information from such data. Compared with other forms of public transportation, ride-hailing data can reveal significantly more private information about millions of riders. For example, the SP and other parties with access to this data (e.g., by agreement, coercion or attack) can determine where riders live, work, socialize, where exactly they go [45, 55, 61], even for one-night stands [49]. Furthermore, there are reported incidents of SP's employees abusing such data to track riders for entertainment [27] or revenge [64].

Provider	SP→R Location tracking	O→D PII harvesting
Uber	○	◐
Ola Cab	○	○
Lyft	○	◐
GrabTaxi	○	◐
EasyTaxi	○	○

Table 2. Robustness level of popular RHSs against high-risk threats (Section 3.2). Current RHSs provide no protection (○) or only partial protection (◐) against these threats.

O→D PII harvesting. A *malicious outsider* could exploit privacy weaknesses in the services offered by the SP, to efficiently collect PII of a large number of drivers. A particular weakness that could be exploited for this purpose is that, in current RHSs, the drivers' information is revealed to the rider *before* the ride begins in order to coordinate the pickup. Hence, an adversary could efficiently collect drivers' PII in a particular area by using fake rider accounts, selecting different pickup locations, requesting and then canceling rides. This attack can be used by shady marketers (e.g., loan sharks) or, even worse, angry taxi-cab drivers trying to physically harm RHSs' drivers [23]. The possibility of this attack was demonstrated when Uber and Lyft employees harvested, from each other, information about thousand of drivers for recruitment purposes [42, 60]. To defend against this attack, the SP could define thresholds on the number of canceled requests that a user can make to the service. If a user passes this threshold, she is banned from the service. However, an adversary could bypass such measures by creating more fake accounts (they are relatively easy to create) or by buying stolen accounts on online black markets [46]. In Section 4, we present an effective solution for this unreported high-risk threat.

3.3 Assessment of Popular RHSs

In this section, we present our experimental evaluation of popular RHSs. In particular, we assess how well these services deal with the two high-risk, RHS-specific threats described in Section 3.2. For this purpose, we installed the

1 Inference attacks against CryptDB [44] do not apply in RHSs as there is no "auxiliary database" available to adversaries.

rider and driver Android mobile app[2] of the selected RHSs and, when possible, made ride requests. In addition, we reviewed the online documentation of the selected RHSs. Table 2 presents the results.

For the *SP→R location tracking* threat, unsurprisingly, we did not find any evidence that the evaluated SPs provide any sort of privacy-preserving mechanisms (e.g., pseudonyms, obfuscated ride traces, ride summaries instead of full location trace) for protecting the privacy and riders from inference attacks. All the evaluated SPs collect the full location trace of each ride, together with the riders' and drivers' identities.

For the *O→D PII harvesting* threat, we determined how much driver information is revealed by the SP when a ride request is made. First, all the SPs reveal the driver's name, phone number, current location, and car plate number; some SPs also reveal the driver's photo and car model and picture. Second, we checked how SPs punish riders if they cancel rides too often, e.g., penalties or lower reputation. Only Uber [59], Grabtaxi, and Lyft [58] provide such types of penalties; this can partially deter harvesting attacks. Reported incidents [4, 42, 60] demonstrate that such mechanisms are insufficient.

4 Our Solution: PrivateRide

We present PrivateRide, our privacy-preserving RHS. PrivateRide is primarily designed to deal with the high-risk, RHS-specific threats presented in Section 3.3, i.e., SP→R location tracking and O→D PII harvesting.

4.1 System Model

We assume a general setting for RHSs; it consists of three parties (Section 2): the riders, the drivers, and the SP. In particular, the SP supports three main operations: (1) ride matching, (2) fare calculation and automatic payment, and (3) reputation ratings for riders and drivers. Moreover, besides localization capabilities, drivers' and riders' smartphones support peer-to-peer wireless communication (e.g., Bluetooth, WiFi Direct). In addition, drivers use a third-party navigation app (e.g., Waze, Google Maps, TomTom) that does not collude with the SP or use an offline navigation app by pre-fetching the map of a large area (e.g., a city or country).

4.2 Adversarial Assumptions

In PrivateRide, we consider the adversarial model presented in Section 3.1. In this model, it is assumed that the SP is *honest-but-curious*, and that it does not act as an enabler for attacks carried out by riders, drivers, or outsiders. As already mentioned, we base these assumptions on the following observations: (1) The SP has incentives to keep its drivers and riders "in check" (and thus avoid threats labeled R→ D or D→ R in Table 1); (2) the SP has incentives to protect the private data of its riders and drivers from outsiders, in order to safeguard its customer-base; and (3) the SP's gains from privacy-breaching attacks against riders or drivers are heavily outweighed by potential loss of business incurred by the public exposure of the SP's acts. If active attacks by the SP are likely to be detected, the SP has incentives to resort to only passive attacks.

The latter observation actually corresponds to the stronger *covert adversary* model introduced by Aumann and Lindell [11]. As an example, consider SP's incentives to provide riders or drivers with a malicious smartphone app that silently collects their data or deviates from established privacy-preserving protocols. As the app would run on all user's phones, a *single* successful reverse-engineering of the app's malicious behavior (e.g., by a competitor) would undoubtedly jeopardize the SP's reputation, consumer trust, hence its entire business model. The resulting business damage is nearly certain in the highly competitive world of RHSs, where it is not uncommon for competitors to attack each other [4, 42, 60]. Therefore, our assumption that an SP will refrain from such malicious behavior and provide users with trustworthy smartphone applications is justified. Extending our full analysis to the covert adversary model is an interesting avenue for future work.

It is also reasonable to assume that the drivers and the SP do not collude with each other, as the drivers are not SP's employees in most RHSs. We also assume that the network and upper-layer protocols do not leak the riders' and drivers' identifiable information to the SP. That is, the SP cannot see the IP addresses of the riders' and drivers' smartphones. This assumption is practical in our system, because most users do not have a direct public IP address. Instead, they are usually behind a NAT offered by the cellular provider. Otherwise, the problem of concealing IP addresses from the SP could also be solved with anonymous network systems, such as Tor, or proxies. Moreover, denial-of-service attacks (DoS) are not considered.

2 We assume RHSs apps offer equivalent privacy and security mechanisms on different mobile platforms.

4.3 Design Goals

This section describes the design goals of PrivateRide. That is, if PrivateRide satisfies these goals, it is robust against the adversarial assumptions described in Section 4.2.

(P1) Rider anonymity. It is computationally difficult for the SP to infer riders' real identities. This requires the anonymity of the riders to be preserved throughout the operations provided by the service, including logins, payments, and reputation rating. This goal addresses the location tracking threat (threat 3 in Table 1) .

(P2) Rider unlinkability. It is computationally difficult for the SP to know whether two rides were hailed by the same rider. This means the unlinkability of the riders has to be preserved throughout the operations provided by the service, including logins, payments, and reputation rating. This goal addresses the location tracking threat (threat 3 in Table 1).

(P3) Harvesting-attack resistance. It is computationally difficult for a malicious outsider to massively collect riders' and driver's PII (e.g., drivers' vehicle information) operating in a region. This goal addresses the PII harvesting threat (threat 6 in Table 1).

(S1) Accountability. The SP is able to revoke the anonymity of misbehaving riders (e.g., a rider who physically attacks a driver). However, the SP does not have full control over this re-identification operation, i.e., it is able to do it only with the support of a driver.

(S2) Secure payment and reputation schemes. Some riders might be more motivated to misbehave and attack the payment and reputation systems, given the anonymity and unlinkability goals. Therefore, it is computationally difficult for an anonymous rider to perform these attacks.

In addition, the following goals guarantee that PrivateRide is usable and economically viable.

(U1) Usability. The system provides approximately the same level of convenience to riders and drivers, as in current RHSs. For instance, riders are not required to perform a significant number of additional steps to hail a ride.

(E1) Economic viability. The system preserves the economic viability of the SP as in current RHSs. That is, the SP can profit by charging a commission on each ride and selling rides' anonymous partial location traces to other parties, e.g., for city planning [15]. The SP can also benefit from partial anonymous ride data to improve its services and capacity planning. In addition, the system is cost-effective to deploy (e.g., compatible with current IT infrastructure).

4.4 Privacy and Cryptographic Tools

PrivateRide uses well-established privacy and cryptographic tools for ride-hailing settings, as briefly introduced below. Such an approach is important because it increases the likelihood of protocol understanding and the acceptance by users and developers, e.g., predictable performance and formal proofs.

Blind signatures. A blind-signature scheme [20] is a form of digital-signature scheme in which the signature requester disguises (blinds) her message before sending it to the signer. The blind signature can in turn be 'unblinded', to obtain a valid signature for the original message. The key property of blind signatures that is important in PrivateRide is that a signer who is asked to verify the signature of an unblinded message is not able to link this message back to the blinded version she signed.

Anonymous credentials. An anonymous credential (AC) is a cryptographic token with which the credential owner can anonymously prove to another party that she satisfies certain properties. In PrivateRide, a user is identified when she obtains ACs, however later, when she wants to start an anonymous session, she anonymously reveals to the SP only the attributes that are needed to be allowed to use the service. PrivateRide relies on the Anonymous Credentials Light (ACL) system [13] because it is computationally efficient and provably secure. However, note that ACL is a linkable anonymous credential scheme, i.e., a user can only use a credential once to avoid her transactions from being linkable.

E-cash. An e-cash scheme (e.g., [21]) enables secure and private electronic payments by providing similar security and anonymity as physical cash. The monetary value is represented by electronic coins that are pieces of data blindly signed by the bank. When a payer deposits a coin to a payee, the payee can check whether the coin has not been deposited before. Properties of e-cash that are important in PrivateRide include payment anonymity (i.e., the bank does not know who is spending the coins) and payment unlinkability (i.e., spendings of a user are not linkable by the bank).

Cloaking. Researchers have shown that removing users' identities from their location-based requests might not be sufficient to protect their anonymity [33]. To prevent this, one of the most popular location-privacy techniques is to cloak users' locations such that k users appear as potential senders of a query, thus achieving *k-anonymity*. To do so, usually, there is a trusted anonymizer who receives location-based requests from users and adaptively blurs their locations into cloaked spatial regions that satisfy the user-specified k-anonymity level. In addition, the

anonymizer can also cloak the time, i.e., delaying location requests such that there are k users within the user-defined time interval and the maximum location cloaking area [65].

We choose a static spatial- and temporal-cloaking approach, as it does not require additional parties and it reduces the chances of fare-calculation errors. It works as follows. For end-point locations of a ride, the SP generates a quantized version of the coordinates (i.e., the SP discretizes the region into a grid of two-dimensional areas). Assuming that a user is located at a point $loc=(x,y)$ in the Euclidean plane, her cloaked location is a tuple $([x_1,x_2],[y_1,y_2])$ that specifies the two-dimensional area where she is located [34]. Hereafter, the precise location of a user is denoted as loc, and the cloaked location of a location point loc is denoted as $\overline{loc}$. For rides' pick-up and drop-off times, the SP discretizes the day into time epochs of a pre-defined value, e.g., 2 minutes. The driver reports to the SP the time intervals where the pick-up and drop-off events occur. For example, if the time interval is 2 minutes, all the pickups occurring between 0:00 and 0:02 would be reported to the SP at 0:02. Thus, pickups in the same cloaked pick-up location and in the same time interval would appear indistinguishable from each other with respect to the SP.

4.5 PrivateRide Overview

In this section, we present an overview of PrivateRide and the rationale behind its design.

Design options. Different privacy-enhancing technologies (PETs) could be used to improve privacy in RHSs. For instance, anonymous matching of riders to drivers by using techniques such as secure multiparty computation [19] or privacy-enhanced matchmaking protocols [40] can be problematic. First, such techniques are complex and could significantly degrade the overall performance of the system. Second, the SP does not learn any information about the ride, thus making accounting difficult, e.g., the SP cannot charge a commission per ride without knowing the distance of the ride. This would also degrade the economic viability of the system, i.e., the SP cannot benefit from the anonymous ride data to improve its services or for capacity planning. Third, providing anonymity to riders and drivers makes accountability challenging, e.g., the SP will not be able to identify and punish misbehaving riders or drivers. Solving this problem often requires a trusted third-party; however, this is not practical for RHSs. Hence, PrivateRide focuses on balancing the trade-off between riders' privacy, the convenience of the service, and SP's benefits by enabling riders to be anonymous during a ride and revealing only partial ride information to the SP (Figure 1).

Design intuition. In PrivateRide, the rider obtains, in advance, ACs and e-cash from the SP. To hail a ride, she first uses an AC to anonymously create a session to the SP, and she sends to the SP her cloaked pick-up location. The SP matches the rider to an available driver whose cloaked location is the closest to the cloaked pick-up location. Thereafter, the driver and rider will establish a secure channel via the SP. This channel is used by the rider and driver to exchange their exact locations and other identifiable information that should not be observed by the SP. When the driver and the rider are in proximity, they use a short-range wireless technology (e.g., Bluetooth) to validate proximity, to prevent O→D PII-harvesting attacks. Upon success, the driver sends her identifying information to the rider, e.g., her profile photo, car registration plate, and then they go about their ride. Ride-start and -end notifications are sent to the SP at predetermined time intervals (e.g., every 2 min). Upon reaching the cloaked drop-off location, the driver signals the end of the ride. The SP receives this notification in the next time interval and proceeds to compute the ride fare and charge the rider. Optionally, the rider and driver anonymously vote for the reputation of each other.

Challenges. The aforementioned design introduces several technical challenges. First, the rider might want to take advantage of her anonymity to misbehave, e.g., she might attempt to underpay for her ride. Second, the anonymous reputation scheme must be *non-monotonic* and it must guarantee that (1) a rider cannot arbitrarily create a reputation token to rate herself or to rate other drivers, (2) the rider cannot fool the driver into rating a different rider, and (3) the SP cannot link a rider with a driver through their reputation operations. To discourage rider misbehavior, PrivateRide requires the rider to deposit to the SP a fixed amount of e-cash when she requests a ride. If the rider misbehaves, the SP can offer her two options: to pay a penalty for the cancellation or to reveal her identity to lower her reputation value. In addition, this approach facilitates automatic payment at the end of the ride. Moreover, to deal with the anonymous reputation challenges, PrivateRide relies on a unique characteristic of RHSs, i.e., the rider and the driver can build a secure channel between them; via this channel, the rider can prove her identity to the driver, and they can exchange resources that should not be seen by the SP. From the exchanged resources, the rider and the driver create and exchange their reputation tokens, so that they can rate each other after the ride. In PrivateRide, a reputation token is a token that enables its holder to anonymously vote for the reputation of the party whose identity is contained in the token.

5 PrivateRide Protocols

This section details the operations of PrivateRide. Table 3 contains the notations we use throughout the description of our solution.

5.1 Ride Prerequisites

At registration time, each rider and driver is issued a digital certificate signed by the SP, denoted as $cert_r$ and $cert_d$, respectively. The certificate contains a public key and an ID pointing to the rider's/driver's registration information. Moreover, to anonymously hail a ride, a rider first needs to acquire ACs and e-cash from the SP.[3] These two operations must be done in advance (i.e., they should not be obtained right before the ride), in order to avoid time-correlation attacks by the SP (i.e., to deanonymize the riders, the SP might try to correlate the AC and e-cash requests with the log-in and deposit events). We now describe these two operations:

Buying e-cash. To buy e-cash, the rider sends a request that contains her real credentials (e.g., certificate or username/password) and the desired amount of e-cash to the SP. The SP sends her the e-cash and charges her, using her registered means of payment (e.g., her credit card). This operation could be automatically done by the rider's app: Some random time after a ride is completed, the rider's app automatically buys e-cash from the SP to maintain a user-defined, but unknown by the SP, amount of e-cash in her local wallet. Random timing prevents time-correlation attacks by the SP; the user-defined threshold of e-cash balance prevents the SP from being able to deanonymize a rider through her deposit amount. The e-cash is issued on fixed denominations to avoid the linking of withdrawing and spending operations.

Obtaining ACs. In PrivateRide, an AC is a cryptographic token that enables its holder to anonymously create a ride session. An AC for a rider r has three attributes: sk_r, rep_r and exp, where sk_r is the secret key associated with the public key in her certificate $cert_r$, rep_r is her reputation at the time the AC is issued, and exp is the coarse-grained expiry time of the AC to prevent the SP from being able to deanonymize the users based on the expiry time of their ACs (e.g., all ACs issued on the same day will expire at the end of that day).

3 Note that some RHSs such as Ola Cab support the use of virtual money [48].

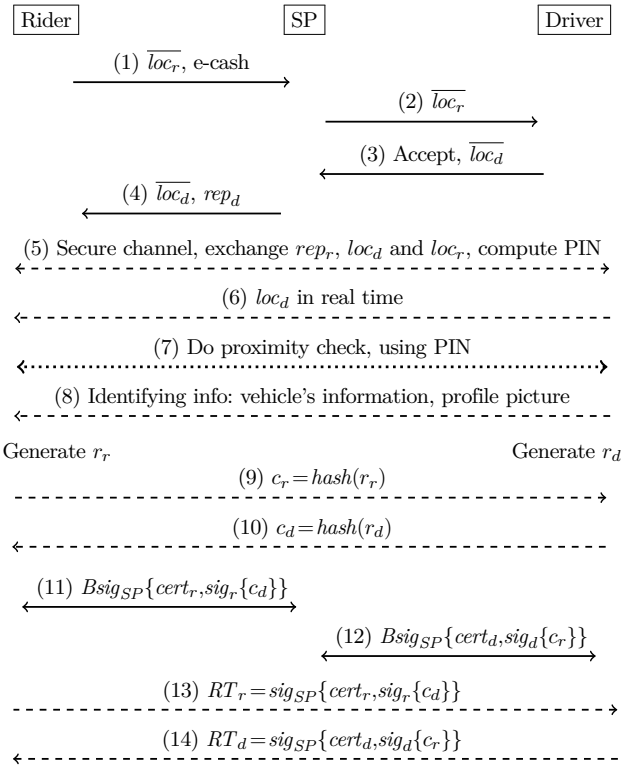


Fig. 3. Initiating a ride request. The dotted arrows represent the proximity channel and the dashed arrows represent the secure channel (via the SP).

Because the selected AC scheme is linkable (i.e., one-time use, otherwise unlinkability is broken), PrivateRide allows riders to obtain several ACs at a time (i.e., to hail multiple rides). However, this approach means that the reputation value included in the ACs might not be up-to-date. To avoid abuse by possible misbehaving riders, the ACs include an expiry date (e.g., one day). In addition, the SP could broadcast a list of riders that have been banned from the service. Based on this revocation list, before the ride starts, the driver can check if the rider is eligible for the service (more details in Section 5.2).

5.2 Ride Initiation

In this section, we explain how a rider and a driver log in to the service. We then describe the actions performed by the rider, the driver, and the SP when a ride request is issued.

5.2.1 Logging In to the Service

Rider. A rider uses her AC to anonymously log in to the service, i.e., she sends to the SP the expiry date exp of her

Notation	Description
r, d	Rider r and driver d
$cert_x$	Public-key certificate of x
rep_x	Reputation of x
$\overline{loc_x}$	Cloaked location of x
loc_x	Precise locations of x
$hash()$	A cryptographic hash function
c_x	Hash challenges send by x
r_x	Random number generated by x
$sig_x\{m\}$	Message m and digital signature of x on message m
$Bsig_{SP}(m)$	Blind signature of the SP on message m
RT_x	Reputation token used to rate x

Table 3. Table of notations for Figure 3

AC (**goals P1 and P2**). In addition, she proves to the SP, in a zero-knowledge fashion, that the claimed value is correct, and that she knows the secret key sk_r that is tied to the AC. If all the proofs are correct, the SP assigns a one-time session ID to the rider, in order to keep track of that session (e.g., for payment and coordination).

Driver. Unlike the rider, the driver logs in to the service non-anonymously by using her real credentials, and she periodically reports her cloaked location $\overline{loc_d}$ to the SP. The driver is also assigned a one-time session ID by the SP.

For the sake of exposition simplicity, in the subsequent protocols, we omit the session IDs in the messages exchanged between the rider or driver and the SP.

5.2.2 Initiating a Ride Request

Once the rider is logged in, she can initiate a ride request to the SP. At this stage, the main purpose of the system is four-fold: (1) the SP matches a rider to a driver, based on their cloaked locations, (2) the driver can verify the proximity of the rider before sending his identifying information to the rider, (3) the rider and the driver securely exchange their private information with each other (e.g., their precise locations, the vehicle's information and the rider's reputation score rep_r), and (4) the rider and the driver create and exchange their *reputation tokens*, so that they can rate each other after the ride. In PrivateRide, a *reputation token* is a token that enables its holder to anonymously vote for the reputation of the party whose identity is contained in the token. The protocol is illustrated in Figure 3.

1. The rider sends her cloaked pick-up location $\overline{loc_r}$ and, optionally, her cloaked destination, to the SP. She also makes a deposit to the SP of a fixed amount of e-cash defined by the SP. The deposit is linked to the session ID of the rider, and it is used as a disincentive for client misbehavior, as will be discussed in Section 5.5.
2. The SP matches the rider to a driver, based on their cloaked locations. Note that this could result in sub-optimal matching between ride requests and offers, as will be discussed in Section 8.3. It then sends to the driver the rider's cloaked pick-up location $\overline{loc_r}$.
3. The driver accepts the request and sends her latest cloaked location to the SP. Otherwise, the driver can decline and the SP can try with different drivers.
4. The SP sends to the rider the cloaked location $\overline{loc_d}$ and the reputation rating rep_d of the driver.
5. The rider and the driver establish a secure channel via the SP e.g., using Diffie-Hellman protocol to exchange data that should not be observed by the SP. From the information used to derive the secret key of the secure channel, they compute a shared secret *pairing PIN*. This *pairing PIN* will be used to prove that the rider is in the proximity of the driver (step 7). Once the channel is established, the rider and the driver use this channel to exchange their exact locations (i.e., loc_r and loc_d, respectively). Using this channel, the rider can also reveal and prove her reputation value to the driver using her AC. The driver can abort the protocol at this step, if the rider's reputation is too low.
6. The driver starts driving from her location loc_d to the pick-up location loc_r, using her third-party or off-line navigation app. She sends, in real time, her precise locations to the rider, via the secure channel, hence the rider can track the movements of the car. Also, at this point, the rider and the driver can call or message each other through their ride-hailing apps, if needed.
7. When the rider and the driver are in proximity, they use a short-range wireless technology (e.g., Bluetooth or WiFi Direct) to set up a proximity channel using the *pairing PIN*.[4] If the channel is successfully established, the driver can verify that the rider is in her proximity before releasing her identifying information to the rider (in step 8). This prevents the harvesting attack described in Section 3.3 (**goal P3**). If this step fails, the driver can decide to abort the protocol.
8. The driver sends to the rider, via the secure channel, her identifying information (e.g., license-plate number and profile picture). This information helps the rider to identify the driver and her car and to prevent

4 As discussed in Section 7, this proximity channel can also be used to detect MITM attacks by the SP in step 5.

certain threats, e.g., fake drivers [25]. Therefore, the required communication distance between the phones of the rider and the driver is small (e.g., several meters). Note that the driver's identifying information is *not* used by the rider to decide whether to accept a ride with the driver, such a decision is made earlier in the protocol (steps 3-4).

In the subsequent steps, the rider and the driver follow a challenge-response protocol over the secure channel to create and exchange their reputation tokens. For the sake of simplicity, when a cryptographic signature on a message m is sent from one entity to another entity, we imply that the original message m is also sent.

9. The rider sends c_r to the driver, where c_r is the cryptographic hash of a random number r_r.
10. Similarly, the driver sends c_d to the rider, where c_d is the cryptographic hash of a random number r_d.
11. The rider creates her reputation token $RT_r = sig_{SP}\{cert_r, sig_r(c_d)\}$, as follows. First, she creates a message consisting of her certificate and her signature on c_d. She then interacts with the SP to obtain a blind signature on the message. This signature and the hash c_d guarantee that a rider cannot arbitrarily create a reputation token and use it to rate herself. In addition, the fact that there is no driver information included in the token provides unlinkability between the rider and the driver (more details in Section 6).
12. As in the previous step, the driver creates her reputation token $RT_d = sig_{SP}\{cert_d, sig_d(c_r)\}$.
13. The rider sends RT_r to the driver. The driver checks the correctness of the SP's signature, whether $cert_r$ is valid and has not been blacklisted by the SP, and whether c_d is correctly signed w.r.t. the public key specified in $cert_r$. If any of the checks fails, the driver can report the failure to the SP, which can take actions accordingly. For example, the SP can charge a penalty to the rider before returning her e-cash back or can request the rider to reveal her identity before returning her e-cash (**goal S2**).
14. The driver sends RT_d to the rider. The rider validates it, as described in the previous step. If any of the checks fail, the rider can report the driver to the SP.

The ride begins once the exchange of reputation tokens between the rider and the driver is completed. The driver starts reporting the exact location to the SP when the car exits the cloaked pick-up area and stops doing so when entering the cloaked drop-off area. The exact pick-up and drop-off times are not revealed to the SP. Instead, the closest time intervals in the future are reported (see Section 4.4). If the car exits the cloaked pick-up area before the next time interval, then the pick-up time is the time interval previous to exiting the cloaked area.

5.3 Ride Termination

The main purpose of this procedure is to enable the SP to charge the rider from her session deposit and pay the driver. First, the SP computes the ride's fare, using the cloaked pick-up and drop-off locations and cloaked times, and the location trace outside of the cloaked areas reported by the driver. Note that this could affect the accuracy of fare calculation, as will be discussed in Section 8.3. Next, based on the one-time session-ID of the rider and the driver, the SP finds the rider's deposit and puts it in the billing information of the driver.

If the rider's anonymous session is still active at the end of the ride, the SP returns to her the remaining e-cash from her deposit. Otherwise, later, the rider can send an anonymous request with her session-ID to be reimbursed. If the deposit is not sufficient to pay for the ride, the SP can ask the rider to deposit more e-cash. If the rider refuses to do so, the SP asks the driver to reveal the rider's certificate $cert_r$ and reputation token RT_r (**goal S2**). With this information, the SP can revoke the rider's anonymity and charge her directly. This situation should not happen frequently, i.e., only when the rider runs out of e-cash in her rider's app. Also, the rider's location privacy is still protected by the cloaking of her pick-up and drop-off locations.

5.4 Reputation Rating

In this section, we describe how the rider and the driver use the reputation tokens that they received during ride initiation (Section 5.2) to rate each other. The reputation rating is optional and asynchronous.

In PrivateRide, it is important to note that, in order to avoid time-correlation attacks by the SP, the rating should not occur right after the ride. This can be implemented by imposing some random delay after the ride before reputation rating for that ride can occur. In addition, the rider and the driver *do not* provide their identifying information to the SP during the reputation rating. That is, the IP addresses and real credentials of the riders and drivers are not known to the SP. This, together with the fact that there is no information of the token holders included in the token, provides unlinkability between the rider and the driver.

Rider. The rider sends to the SP the reputation token $RT_d = sig_{SP}\{cert_d, sig_d(c_r)\}$ received from the driver, the random number r_r used to generate the hash c_r, to-

gether with the ride's rating. When the SP receives this token, it checks the correctness of the signature, the correctness of the certificate $cert_d$, whether c_r has not been used before and whether c_r matches with the hash of r_r.

Driver. The driver sends to the SP the reputation token $RT_r = sig_{SP}\{cert_r, sig_r(c_d)\}$ of the rider, the random number r_d used to generate the hash c_d, together with the ride's rating. The SP does the same checks as described above for the rider.

5.5 Ride Cancellation

As in current RHSs, both the driver and rider can cancel the ride at any point. However, ride cancellation is discouraged by the SP, because it can lead to malicious behavior from the rider and the driver. Therefore, similarly to current RHSs, in PrivateRide, if riders or drivers cancel a ride a certain amount of time after the ride request, they are punished by the SP, e.g., their reputation scores are lowered or fees are charged . For example, a Uber rider would be charged a $10 cancellation fee if she cancels more than 5 minutes after her ride request [18]. If the driver cancels a ride, the SP can punish the driver directly, as her identity is not hidden. If the rider cancels a ride before the reputation tokens are generated (step 11 in Figure 3), the SP can offer the rider two options: pay a penalty for the cancellation or reveal her identity and have her her reputation value lowered. This offers a trade-off to deter misbehaviors. If the rider cancels the ride after the reputation tokens are generated, then the driver will lower the rider's reputation using the token (without revealing the identity of the rider to the SP). The driver should also invalidate his reputation token by revealing it to the SP. As stated before, ride cancellations should not occur frequently and even when the rider anonymity is revoked, her location privacy is still protected by the cloaking of her pick-up and drop-off locations.

6 Privacy Analysis

In this Section, we present our analysis of PrivateRide to show it achieves its privacy goals, i.e., P1, P2, and P3 in Section 4.3, w.r.t. different assumptions about the side information known by the SP.

Dataset. To assess the performance of our solution, we rely on a dataset of NYC taxi rides in New York [63]. This dataset provides the pick-up and drop-off locations and times, the total fare, and information about the drivers. We selected all the valid rides from a random weekday (e.g., rides with reported distance higher than zero). From this set, we chose rides of which the pickups are from the boroughs of Manhattan. This results in 489,479 rides with length of 2.4 $\pm$ 2.6 km. These boroughs are chosen due to their high density of places of interests and activities, compared to other boroughs, hence the locations of people would be more sensitive.

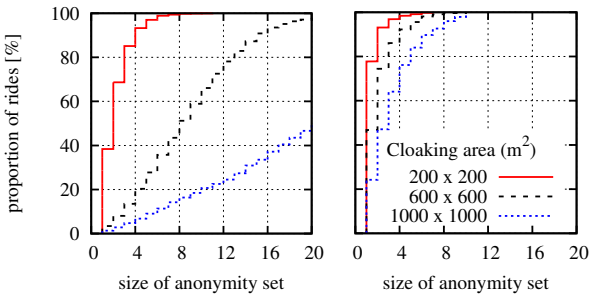
6.1 SP $\rightarrow$ R location tracking (goals P1, P2)

PrivateRide guarantees rider's anonymity and unlinkability throughout its operations, as we explain next.

Log-in and payment. Provided that a rider uses two different ACs for two login sessions, the SP cannot link the two login sessions with each other. It also cannot deanonymize a user through her anonymous login using her AC. These are guaranteed due to the anonymity and unlinkability properties of the ACL anonymous credential scheme [13]. The SP cannot deanonymize or link different rides of a rider through the e-cash tokens that she deposits, due to the anonymity and unlinkability guarantees of e-cash [16, 17, 21]. The SP also cannot deanonymize a user based on the amount of her e-cash deposit, because the deposit amount is fixed and defined by the SP; from the NYC dataset mentioned above, if the fixed deposit amount is set to $20, it would cover 90% of the rides. Moreover, the SP cannot deanonymize a past ride of a rider based on the time she buys e-cash and/or the amount that she buys, because the e-cash acquisition operation occurs at a random time after a ride finishes and the e-cash balance threshold set by the rider app is not known by the SP.

The SP, however, might try to make time-correlation attacks to break the anonymity of riders (by correlating AC and e-cash requests with login and deposit events), because the SP serves as the issuer of ACs and e-cash, and the riders are identified when they obtain these resources. That is, if a rider obtains an AC and e-cash, and in a short time interval, she logs in and makes a deposit to the SP. If in that time interval, there is only one login event, the SP would be able to link that ride with the rider who just obtained the AC (i.e., the rider's anonymity is broken). Given the fact that recently, Uber outnumbers taxis in NYC [5], and from the NYC dataset, there are, on average, respectively, 530 and 47 ride requests per minute, during peak hour and least-busy hour, a rider would have a good anonymity set of size at least 47 if the time gap between her AC and e-cash request and AC and e-cash deposit is at least a minute.

Reputation rating. The SP might try to deanonymize a rider by looking at the reputation tokens that have been



(a) 7 PM - 8 PM **(b)** 4 AM - 5 AM

Fig. 4. Evaluation of the anonymity set of a ride for time interval of 2 minutes and different sizes of the cloaking areas, at least-busy hour (right) and at peak hour (left) in a random weekday in Manhattan.

used to vote for her and guessing the rides where these tokens were generated. However, it is computationally difficult for the SP to do so because, when a reputation token is generated (during ride initiation), it is blindly signed by the SP, hence the SP cannot link the unblinded token with the blinded token it signed before, as guaranteed by the blindness property of blind signature schemes [20]. In addition, the unblinded reputation token is sent over a secure channel established between the rider and the driver, hence not readable by the SP. A concern could be that the SP might try to find the set of rides taken by a rider by guessing the identities of the drivers who rate for that rider's reputation. This, however, is not possible because the information about the drivers is not included in the rider's reputation tokens and the drivers do not provide any identifying information to the SP when they rate a rider. In addition, given two reputation tokens that are used to rate the same rider, the SP cannot link together the two rides where the tokens were generated, because the reputation tokens do not contain any information about the rides or any information generated by the SP, other than the SP's signature on the tokens.

End-point location and time cloaking. The level of privacy protection offered by location and time cloaking depends on the amount of information the SP has about the riders. In an extreme case, the SP might know the identity, as well as the exact or approximated location and time when a particular rider hailed a ride. The SP can use this information to link the rider's identity to an anonymous ride in its database and determine the rider's destination. For instance, the SP has a photo of a celebrity hailing a ride in the street and it wants to learn the celebrity's destination [55]. In such a situation, the protection offered by PrivateRide is proportional to the number of rides that occur in the same cloaked area and time interval, i.e., the size of the anonymity set.

To estimate the anonymity set of a ride in PrivateRide, we use the NYC dataset described earlier. Figure 4 shows the experimental cumulative distribution function of the anonymity set for time intervals of 2 minutes and three different sizes of the cloaked pickup: 200 m × 200 m, 600 m × 600 m and 1 km × 1 km, at the peak hour (from 7 PM to 8 PM) and the least-busy hour (from 4 AM to 5 AM) of the day. The results show that for cloaking areas of size 600 m × 600 m, a ride would have a good anonymity set: During the peak hour, 80% of the rides would have an anonymity set of at least 4 and 40% of rides would have an anonymity set of at least 10; and during the least-busy hour, 50% of the rides would have an anonymity set of at least 2. As expected, a larger cloaking area provides more privacy for the riders, e.g., with cloaking size of 1000 m × 1000 m, during least-busy hour, nearly 80% of the rides would have an anonymity set of at least 2, and during the peak hour, 80% of the rides would have an anonymity set of at least 10.

The anonymity set is likely to be lower in areas with less commercial activity and population (e.g., residential areas); however, we need to take into account that this is strong and non-scalable attack that is unlikely to affect most riders. Still, PrivateRide, in contrast with current RHSs and traditional taxi services, provides some level of protection against such attacks. In the worst case scenario of rural areas where there is only a single house in a cloaked area, it is likely that ride requests coming from that area belong to people living in that house. However, the attacker still has a certain uncertainty, because the requests might also come from other people, e.g., visitors. Also note that people living in such areas are less likely to rely on RHSs/taxis for they daily activities and rides in such cases are likely to have lower privacy implications.

In general, the SP knows only some basic information about most riders. Similarly to other location-based services, a reasonable assumption for RHSs is that the SP knows publicly available information about riders' homes and work places [33]. Thus, rides between a rider's home and work are generally easier to link and deanonymize than other rides. Nevertheless, the former normally reveal less sensitive information about the rider. Of more interest for the SP or other adversaries, are rides that are less frequent and to different type of destinations. With PrivateRide, if the SP wants to learn the destinations of the rides starting from home or work of a particular rider, without any additional side information, the size of the anonymity set in this case is *at least* equal to the number of riders living or working in the same cloaked area as the targeted rider (the size is larger if we consider riders temporarily visiting the cloaked area and hailing rides). Unfortunately, due to the lack of datasets about the geographic distribution of RHS' riders,

we could not quantify the privacy gain for these cases. This is a limitation in our work. Nevertheless, it is easy to see that, in general, the size of the anonymity set should be significantly larger than in the previously described attack where the SP knew the exact time and location of a ride. Similarly, large-scale inference attacks, e.g., to learn where riders go shopping [45], clubbing, or to learn which riders have one-night stands [49], are more difficult for the SP without additional information to determine the exact pick-up and drop-off locations or to link the observed rides.

Note that the results presented in this section are the *lower-bound* privacy gain, because a rider does not have to do anything in order to obtain this level of protection. A more privacy-conscious rider can have better privacy protection, if, instead of always hailing a ride from the same cloaked pick-up location, she uses different neighboring cloak areas or she selects busier cloaking areas. Also, as discussed in Section 4.4, a trusted third-party tool could be used to suggest riders what nearby cloaked pick-up areas provide better privacy based on the rider's ride history and the ride density in each cloaked area.

6.2 O → D PII harvesting (goal P3)

In the current form of RHSs, a malicious outsider can easily harvest PII from drivers: it could create a rider account, fake different pick-up locations, make ride requests, obtain drivers' information and then cancel the ride requests. This is possible, because the drivers' information is revealed to the riders right after the riders are matched with the drivers. With PrivateRide, the driver can easily check the proximity of the rider by using a short-range wireless communication channel before sending her PII to the rider. Thus, to collect PII from drivers, a malicious outsider not only needs to make a ride request but also to be physically close to the pick-up location reported (see Section 8 for performance evaluation of this operation).

7 Security Analysis

We analyze PrivateRide to show that it maintains the same level of security as current RHSs: Miscreants can be punished when they misbehave **(goal S1)**; the riders and drivers cannot lie about their reputation scores; and they cannot cheat on the commission of the rides **(goal S2)**. In this analysis, we assume that riders and drivers might attempt attacks against the integrity of the system. For example, riders might attempt to underpay for their rides or unduly increase their reputation; drivers might attempt to overcharge riders or unduly increase their reputation. In addition, we show that *active attacks* mounted by the SP can be detected by riders and drivers with minor modifications.

Riders. A rider might want to arbitrarily increase her reputation. First, during ride initiation, a dishonest rider can attempt to cheat by claiming a better reputation for herself. However, this would be detected by the driver, because the proof for attributes in her AC will not be correct w.r.t. her falsely claimed reputation [13].

A dishonest rider might also want to arbitrarily create a reputation token and rate herself, in order to improve her reputation. This attack is prevented because a rider cannot obtain a signature from the SP on her reputation token if she is not associated with an on-going ride. A dishonest rider might also want to prevent the driver from rating her, as follows: After generating her reputation token, she uses it to rate herself before sending it to the driver (i.e., when the driver uses the token to rate the rider, the SP would detect that the token had been used hence refuse to accept the rating). PrivateRide prevents this by requiring the rider, during the reputation-rating protocol, to reveal the random number r_d that was used as the input of the cryptographic hash function to generate the hash challenge (c_d) included in the reputation token. Assuming the one-way property of the cryptographic hash function holds, it is computationally impossible for the rider to find this r_d, hence her cheating attempt fails. Another concern could be that a dishonest rider might want to avoid being rated by faking her identity in her reputation token (so that the rating would go for someone else). This attack is prevented, because when the driver receives the reputation token from the rider, the driver can check if the signature on the hash challenge c_d that she sent to the rider is correctly signed w.r.t. the public key in the certificate claimed by the rider.

Drivers. A dishonest driver might want to unduly create a reputation token and rate herself, in order to improve her reputation. This attack is prevented, with the same reasoning as presented above for the riders, because the rider and the driver follow the same procedure for voting for the reputation of each other. A dishonest driver might want to overcharge a rider, by e.g., reporting to the SP a longer route for the ride. PrivateRide offers the same guarantees against this overcharging attack, as in current RHSs, because they both rely on the information reported by the driver's phone; in future work, we will explore mechanisms to protect against this type of attacks.

A dishonest rider and a dishonest driver might want to collude with each other, in order to falsely increase the reputation scores for both of them: the rider makes a ride

request with her pick-up location close to the location of the driver, so that they are assigned to each other by the SP. Then they can have valid reputation tokens and use them to give good ratings for each other. This attack, however, is monetarily expensive, hence not scalable, because each trip is charged a commission by the SP. A dishonest rider and a dishonest driver might want to cancel the ride earlier than its actual end, in order to underpay for the commission of the ride. This, however, can be prevented by PrivateRide, because when a rider or a driver cancels a ride, the SP will punish them by lowering their reputation according to its service policies. The driver is identified during the ride, hence this punishment operation is straightforward. For the rider, PrivateRide enables this punishment operation by allowing the SP to revoke her identity, i.e., it requests the misbehaving rider to reveal her identity, in order to get her deposit reimbursement.

The SP. PrivateRide was presented under the assumption of an *honest-but-curious* SP, however, with some minor modifications, it can be resistant to some *active attacks* performed by the SP. First, a *malicious* SP might want to perform man-in-the-middle (MITM) attack (at step 5), in order to get the precise pick-up location of the rider and her reputation score. This can be easily detected by the rider and the driver: Via the proximity channel, the rider and the driver can cross-check that the tokens they exchanged through the SP to establish the secure channel were not altered (i.e., the SP did not perform a MITM attack). Second, a *malicious* SP might want to fool a driver into revealing the identity of the rider who rides with him: the SP could tell the driver that the rider did not deposit enough for the ride, hence it needs to revoke the rider's anonymity, in order to charge her directly from her registered billing information. This, however, can be easily prevented, by allowing the rider to get a deposit certificate from the SP each time she deposits. The rider can send her deposit certificate to the driver to prove to the driver that she pays enough for the ride. Note that in practice, for the sake of its reputation, a SP would certainly not take the risk of being caught carrying out such an attack.

8 Implementation and Evaluation

In this section, we experimentally evaluate PrivateRide's performance overhead. We use real data-sets to estimate PrivateRide's effect on fare calculation and ride matching.

8.1 Implementation Details

PrivateRide's prototype includes the main cryptographic operations needed for requesting a ride (i.e., AC operations, blind signatures) and for setting up the proximity channel between two phones. For the server, we implemented PrivateRide as an HTTP server in Python by using the Flask framework [26]. For ACL operations, we rely on the petlib library [6]. For RSA blind and standard signatures, we use the Python Cryptography Toolkit (pycrypto) library [54]. For the client, we implemented PrivateRide as an Android application with support for Android v4.4 (SDK 19) and later versions. For ACL operations, we ported the ACL client classes from the petlib library to Java. For RSA blind- and standard- signature support we use the Spongy Castle library [56]. Because Spongy Castle's implementation of elliptic curves (EC) modular multiplications is not efficient in mobile devices, we implemented EC modular multiplications in native C code by using a cross-compiled version of the OpenSSL library v1.0.1r and the Android NDK [24]. For the proximity channel, we implemented both Bluetooth and WiFi Direct technologies.

8.2 System Performance

Experimental Setup. We used PrivateRide's prototype to estimate the overhead added by the most complex cryptographic operations in the protocols used to hail a ride (Section 5). For this evaluation, we used a server (8x3.5GHz, 16 GB RAM) with Ubuntu 12.04 (kernel 3.2) and a smartphone LG G3 (4x2.5 GHz, 2 GB RAM) with Android 5.0. To make our scenario more realistic, the smartphone communicated with the server over a 4G cellular network connection. We considered two sets of security parameters. First, ACs with an EC group of 521 bits (OpenSSL's secp521r1), a value recommended for 128 bits security[5] [13], and 4096 bits RSA keys for blind signatures. However, such level of security might not be required in most RHSs scenarios. Thus, we also considered a slightly weaker set of parameters, probably more appropriate for RHSs: an EC group of 224 bits (value recommended by NIST for EC encryption [9]) and 2048 bits RSA keys. We ran 100 experiments per measurement and report hereunder average values with 95% confidence intervals.

Cryptographic Operations. The total time to obtain an AC credential was 554.2 ± 21.8 and 962.8 ± 20.8 ms for

5 ACL's authors recommend an EC group of 576 bits but OpenSSL's largest secure EC group has 521 bits (secp521r1).

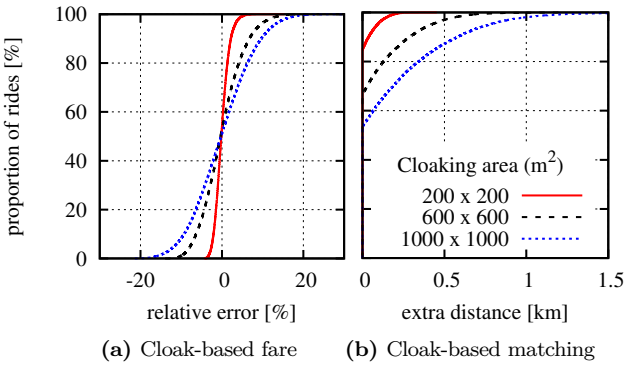
the basic (EC 224 bits) and strong (EC 521 bits) security parameters respectively. Logging in with the AC took in total around 218.9±4.9 ms for the basic configuration and 558.8 ± 23.7 ms for the strong configuration. Blind signatures, as expected, are significantly more efficient: 67.8±2.1 and 71.0 ± 1.9 ms for the basic and the strong configurations, respectively. Therefore, logging in and generating the reputation token will add around 600 ms to ride hailing operations for the strong configuration, which is unlikely to be noticed by a rider. Furthermore, buying and spending e-cash is also practical. For example, assuming that we use ACL for e-cash [37] and an EC group of 224 bits, obtaining a e-cash token will take around 500 ms and spending it around 200 ms. This overhead is practical, especially if we take into account that buying e-cash should be done in advance to requesting a ride.

Proximity Check. We implemented an app for setting up short-range communication channels between two phones. The app supported two technologies: Bluetooth and WiFi-Direct. For each technology, we measured the time needed to set up the channel in two scenarios: (1) a phone is inside a car and the other phone is in open space, and (2) both of the phones are in open space. Our experiments show that, for both scenarios, on average, setting up the proximity channel takes only around 6.4 s for Bluetooth and around 4.5 s for WiFi Direct. Regarding the maximum communication distance between two phones, for the second scenario, it can be as far as 30 m and 50 m, for Bluetooth and WiFi Direct, respectively. This distance, for both technologies, decreases to 15 m for the first scenario.

The supported distance is adequate for our scenario, because this operation is expected to occur when the rider is waiting outside, just before the driver arrives, at the precise pick-up location agreed with the driver (Fig.3, step 5). Moreover, its effect on usability is negligible, as it can be executed automatically in the background with little effort from riders and drivers. Note that the feasibility of the short-range communication channel is also reported in other works, e.g., Murphy et al. [43] show even more optimistic results; they show that it would take less than 2 s to establish a Bluetooth channel between two Bluetooth-enabled devices.

8.3 Effect of Location Cloaking

We evaluate the effect of location cloaking on the performance of PrivateRide, in terms of the error induced on the fare calculation and the resulting sub-optimality of the ride matching. We do so for three different sizes of the cloaking areas: 200 m × 200 m, 600 m × 600 m, and 1000 m × 1000 m; this provides different levels of location privacy.



(a) Cloak-based fare **(b)** Cloak-based matching

Fig. 5. Effect of location cloaking for different sizes of the cloaking areas, on fare calculation (left) and on ride matching (right).

Datasets. To assess the performance of our solution, we rely on two data-sets: a data-set of GPS traces of Uber black cars rides in San Francisco [32] (the SF data-set) and a data-set of NYC taxi rides in New York [63] (the NYC data-set). The SF data-set consists of 22,570 rides with an average length of 2.1±1.9 km. Note that the rides are short because the beginning and the end of the GPS traces were truncated in the published data-set. We considered only rides that were longer than 500 m and that lasted for at least 2 min. The NYC data-set was described in Section 6; this dataset does not provide the GPS traces of the rides, however, the pick-up and drop-off locations and time are provided. Based on these specifics of the two data-sets, we used the SF dataset to assess the effect of location cloaking on fare calculation, and the NYC dataset to assess the performance of ride matching.

Performance of Fare Calculation. As explained in Section 5.3, in PrivateRide, the SP computes the fare of the ride, based on the cloaked pick-up and drop-off locations, the trace outside of the cloaks and the duration of the ride, as reported by the driver. This forces the SP to define a cloak-based fare-computation method; of course, the fare computed in this way should be close to the fare computed with the original method of RHSs. Intuitively, a reasonable estimation of the distance covered inside the cloak is a quantity proportional to the time interval and the perimeter of the cloaking area. We determined the optimal coefficient of proportionality by performing a linear regression with least square error between the ground truth (i.e., the fare calculated when the full trace is known) and the cloak-based method. Note that this pricing method is somewhat similar to the zone-based fare system used by public transport authorities.

Figure 5a shows the experimental cumulative distribution function of the relative error on the cloak-based fare method (compared to that computed by Uber); negative errors indicate that the rider is undercharged (i.e., pays less) for the corresponding ride. The results show that the cloak-based fare method achieves good performance: for cloaking areas of size 600 m × 600 m, 95 % of rides would have a relative error of less than 10%, and, as expected, the smaller the cloaking areas are, the better the cloak-based fare method performs. In addition, the average of the relative errors is close to 0 (with a small variance); therefore, as the number of rides taken by the same rider/driver increases, the fare errors quickly average out to 0.

Performance of Ride-Matching. In order to match ride requests to ride offers, based on the cloaked locations reported by the riders and the drivers, the SP defines a metrics for measuring the distance between two cloaked areas. The ride-matching based on this metric should be close to the optimal solution (i.e., when cloaking is not used, the SP assigns to a rider the driver who is the closest to the pick-up location). In PrivateRide, we use the straight-line distance between the centers of the two cloaking areas (note that more advanced metrics could be used, e.g., by taking into account the road network and the traffic conditions).

Figure 5b shows the experimental cumulative distribution function of the relative extra distances the drivers who are chosen by the cloak-based matching method have to drive. The relative extra distance is computed w.r.t. the distance covered by the driver chosen by the optimal matching method in which cloaking is not used. This value reflects the extra costs for both the driver (gas and driving time to pickup) and the rider (waiting time at pickup). It can be observed that the extra pick-up distance is small: In 78% of the cases, the overhead is less than 0.1km, for the cloaking areas of size 600 m × 600 m.

9 Related Work

We found only one prior work directly related to RHSs, by Chen et al [22]. The authors investigate the effect of surge pricing on Uber users and identify implementation details of Uber's surge price algorithm. However, the work does not discuss or analyze privacy aspects of Uber.

In the line of privacy-enhancing technologies for RHSs, prior works include privacy-preserving solutions for car-pooling services, public transportation services and distance-based services. Friginal et al. [30] propose a distributed solution for privacy-preserving car-pooling services. However, car-pooling is different from ride-hailing in that, in car-pooling, drivers offer rides to people along the route they already plan to travel, whereas RHSs enable people to use their own cars as taxis. Heydt-Benjamin et al. [36] proposed one of the first cryptographic frameworks for transport services. They discuss the challenges that a privacy-preserving transit system should solve. Later, m-ticketing solutions were proposed, such as [10, 38]. Note that the pick-up and drop-off locations in public transportation services (i.e., bus or train stops) are more coarse-grained than in RHSs.

Privacy-preserving solutions for location-based services that charge users based on the distance that they drive include solutions for electronic toll pricing (e.g., [12, 41, 52]) and pay-as-you-drive car insurance (e.g., [62]). These solutions rely on tamper-evident devices installed in the vehicles and random spot-checks to compute the total distance traveled by the vehicles. Later on, Pham et al. [51] proposed a solution based on the networks of access points, to provide a verifiable lower-bound of the distance reported by a user.

10 Conclusions

In this paper, we have analyzed the privacy threats in the current form of RHSs. We have also proposed PrivateRide, a practical solution that enhances location privacy for the riders w.r.t. the SP and privacy for the drivers w.r.t. malicious outsiders, while preserving the convenience and functionality offered by the current system.

The experimental evaluation on our PrivateRide prototype implementation of the rider's app and the ride-hailing server shows that PrivateRide introduces a negligible delay for the cryptographic operations and for the proximity-check operation. Our analysis on the real data-set of rides shows that PrivateRide guarantees good location privacy for the riders. This is achieved with only minor effect on the service's usability, e.g., 95 % of the rides would have a relative error on fare calculation of less than 10%. From an economic perspective, PrivateRide can offer SPs a competitive advantage not only over other RHSs but also over conventional taxi services, as those services also collect riders' PII for reservations and business analytics purposes. As such, this work lays the foundation for the design of a privacy-preserving and secure RHS.

As part of future work, we plan to (1) enhance the design to be resistant to more attacks defined in our threat taxonomy, (2) formally analyze the privacy and security properties of PrivateRide, and (3) investigate cases of collusion between drivers and the SP, e.g., if some drivers work for the SP or the SP makes use of driverless cars.

References

[1] http://rideshareapps.com/2015-rideshare-infographic/. Last visited: May 2016.

[2] http://www.dailydot.com/technology/uber-female-driver-harassment/. Last visited: May 2016.

[3] http://www.reuters.com/article/uber-tech-lyft-probe-exclusive-idUSKBN0U12FH20151219. Last visited: May 2016.

[4] http://www.bbc.com/news/business-35888352. Last visited: May 2016.

[5] http://www.engadget.com/2015/03/18/uber-outnumbers-taxis-in-nyc/. Last visited: May 2016.

[6] https://github.com/gdanezis/petlib. Last visited: May 2016.

[7] http://www.businessinsider.com/blake-jareds-50000-uber-credit-free-rides-for-life-2014-4. Last visited: May 2016.

[8] http://fortune.com/2015/03/30/uber-stolen-account-credentials-alphabay/. Last visited: May 2016.

[9] Cryptographic key length recommendation. http://www.keylength.com/en/. Last visited: May 2016.

[10] G. Arfaoui, J.-F. Lalande, J. Traoré, N. Desmoulins, P. Berthomé, and S. Gharout. A Practical Set-Membership Proof for Privacy-Preserving NFC Mobile Ticketing. *Proc. of the 15th Privacy Enhancing Technologies Symposium*, 2015.

[11] Y. Aumann and Y. Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. In *Theory of cryptography*. Springer, 2007.

[12] J. Balasch, A. Rial, C. Troncoso, B. Preneel, I. Verbauwhede, and C. Geuens. PrETP: Privacy-Preserving Electronic Toll Pricing. In *Proc. of USENIX Security Symposium*, 2010.

[13] F. Baldimtsi and A. Lysyanskaya. Anonymous credentials light. In *Proc. of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013.

[14] http://www.bloomberg.com/news/articles/2015-06-28/one-driver-explains-how-he-is-helping-to-rip-off-uber-in-china. Last visited: May 2016.

[15] www.bostonglobe.com/business/2015/01/13/uber-share-ridership-data-with-boston/4Klo40KZREtQ7jkoaZjoNN/story.html. Last visited: May 2016.

[16] S. Brands. Electronic cash systems based on the representation problem in groups of prime order. *Proc. of the 13th Cryptology Conference*, 1993.

[17] J. Camenisch, J. Piveteau, and M. Stadler. An efficient payment system protecting privacy. *Proc. of the 9th European Symposium on Research in Computer Security*, 1994.

[18] https://newsroom.uber.com/updated-cancellation-policy/. Last visited: Nov. 2016.

[19] R. Canetti. *Studies in secure multiparty computation and applications*. PhD thesis, The Weizmann Institute of Science, 1996.

[20] D. Chaum. Blind signatures for untraceable payments. In *Proc. of the 3rd Cryptology Conference*, 1983.

[21] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proc. of the 10th Cryptology Conference*, 1990.

[22] L. Chen, A. Mislove, and C. Wilson. Peeking Beneath the Hood of Uber. In *Proc. of the ACM Conference on Internet Measurement Conference*. ACM, 2015.

[23] http://www.cnet.com/news/taxi-dispute-gets-physical-in-france-with-attack-on-uber-car/. Last visited: May 2016.

[24] https://developer.android.com/tools/sdk/ndk/index.html. Last visited: May 2016.

[25] http://nypost.com/2016/09/10/fake-uber-drivers-are-scamming-tourists-at-us-open/. Last visited: Nov. 2016.

[26] http://flask.pocoo.org/. Last visited: May 2016.

[27] http://www.forbes.com/sites/kashmirhill/2014/10/03/god-view-uber-allegedly-stalked-users-for-party-goers-viewing-pleasure/. Last visited: May 2016.

[28] www.forbes.com/sites/ellenhuet/2014/09/08/uber-lyft-cars-arrive-faster-than-taxis/#3f819c3c5f73. Last visited: May 2016.

[29] http://www.forbes.com/sites/ronhirson/2015/03/23/uber-the-big-data-company/. Last visited: May 2016.

[30] J. Friginal, S. Gambs, J. Guiochet, and M.-O. Killijian. Towards privacy-driven design of a dynamic carpooling system. *Trans. on Pervasive and Mobile Computing*, 2014.

[31] https://gigaom.com/2014/11/21/if-youre-worried-about-uber-and-privacy-dont-forget-lyft-and-sidecar/. Last visited: May 2016.

[32] https://github.com/dima42/uber-gps-analysis/blob/master/gpsdata/. Last visited: May 2016.

[33] P. Golle and K. Partridge. On the anonymity of home/work location pairs. In *Proc. of the Conference on Pervasive Computing*, pages 390–397. Springer, 2009.

[34] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of the conference on Mobile systems, applications and services*. ACM, 2003.

[35] http://www.theguardian.com/technology/2015/may/23/us-investigates-phantom-cab-rides-on-british-uber-accounts. Last visited: May 2016.

[36] T. S. Heydt-Benjamin, H.-J. Chae, B. Defend, and K. Fu. Privacy for public transportation. In *Proc. of the 6th Privacy Enhancing Technologies Symposium*, 2006.

[37] G. Hinterwälder, C. T. Zenger, F. Baldimtsi, A. Lysyanskaya, C. Paar, and W. P. Burleson. Efficient e-cash in practice: NFC-based payments for public transportation systems. In *Proc. of the 13th Privacy Enhancing Technologies Symposium*, 2013.

[38] A. P. Isern-Deyà, A. Vives-Guasch, M. Mut-Puigserver, M. Payeras-Capellà, and J. Castellà-Roca. A secure automatic fare collection system for time-based or distance-based services with revocable anonymity for users. *The Computer Journal*, 2013.

[39] http://jetsettershomestead.boardingarea.com/2015/01/08/ways-passengers-can-cheat-uber/. Last visited: May 2016.

[40] M. Li, N. Cao, S. Yu, and W. Lou. Findu: Privacy-preserving personal profile matching in mobile social networks. In *Proc. of the Conference on Computer Communications*. IEEE, 2011.

[41] S. Meiklejohn, K. Mowery, S. Checkoway, and H. Shacham. The Phantom Tollbooth: Privacy-Preserving Electronic Toll Collection in the Presence of Driver Collusion. In *Proc. of the 20th USENIX Security Symposium*, 2011.

[42] http://money.cnn.com/2014/08/11/technology/uber-fake-ride-requests-lyft/. Last visited: May 2016.

[43] P. Murphy, E. Welsh, and J. P. Frantz. Using bluetooth for short-term ad hoc connections between moving vehicles: a feasibility study. In *Vehicular Technology Conference, 2002. VTC Spring 2002. IEEE 55th*, volume 1, pages 414–418. IEEE, 2002.

[44] M. Naveed, S. Kamara, and C. V. Wright. Inference attacks on property-preserving encrypted databases. In *Proc. of the ACM Conference on Computer and Communications Security*, 2015.

[45] http://newsroom.uber.com/2014/09/inferring-uber-rider-destinations/. Last visited: May 2016.

[46] https://news.yahoo.com/warning-uber-account-might-sale-black-market-164144470.html. Last visited: May 2016.
[47] http://www.newsweek.com/uber-taxi-e-hailing-riding-app-travis-kalanick-emil-michael-josh-mohrer-uber-285642. Last visited: May 2016.
[48] https://www.olamoney.com/. Last visited: May 2016.
[49] http://www.oregonlive.com/today/index.ssf/2014/11/sex_the_single_girl_and_ubers.html. Last visited: May 2016.
[50] www.owasp.org/index.php/OWASP_Risk_Rating_Methodology. Last visited: May 2016.
[51] A. Pham, K. Huguenin, I. Bilogrevic, and J.-P. Hubaux. Secure and Private Proofs for Location-Based Activity Summaries in Urban Areas. In *Proc. of the 16th ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2014.
[52] R. A. Popa, H. Balakrishnan, and A. J. Blumberg. VPriv: Protecting Privacy in Location-Based Vehicular Services. In *Proc. of the 18th USENIX Security Symposium*, 2009.
[53] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *Proc. of the ACM Symposium on Operating Systems Principles*, 2011.
[54] https://pypi.python.org/pypi/pycrypto. Last visited: May 2016.
[55] http://research.neustar.biz/2014/09/15/riding-with-the-stars-passenger-privacy-in-the-nyc-taxicab-dataset. Last visited: May 2016.
[56] https://rtyley.github.io/spongycastle/.
[57] http://sfist.com/2014/07/30/uber_still_illegally_working_sfo_al.php. Last visited: May 2016.
[58] http://thehub.lyft.com/blog/2014/10/15/cancelations-join-acceptance-rate-equation. Last visited: May 2016.
[59] http://therideshareguy.com/uber-deactivated-a-bunch-of-drivers-as-an-intimidation-tactic/. Last visited: May 2016.
[60] http://www.thewire.com/technology/2014/08/uber-accused-of-booking-thousands-of-fake-rides-with-rival-lyft/375936/. Last visited: May 2016.
[61] http://toddwschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-vengeance/. Last visited: May 2016.
[62] C. Troncoso, G. Danezis, E. Kosta, J. Balasch, and B. Preneel. Pripayd: Privacy-friendly pay-as-you-drive insurance. *Trans. on Dependable and Secure Computing*, (5), 2011.
[63] https://uofi.app.box.com/NYCtaxidata. Last visited: May 2016.
[64] http://www.usatoday.com/story/tech/2014/11/19/uber-privacy-tracking/19285481/. Last visited: May 2016.
[65] M. Wernke, P. Skvortsov, F. Dürr, and K. Rothermel. A classification of location privacy attacks and approaches. *Personal and Ubiquitous Computing*, 2014.
[66] http://wspa.com/2016/01/18/uber-driver-off-the-job-after-he-charged-for-fake-puke-2/. Last visited: May 2016.

A Integrity threats

Table 4 presents a taxonomy for integrity threats in RHSs.

Integrity Threats			
Description			Risk
1) R→ D		fare undercharging [39]	Low
2) R→ SP		reputation cheating	Low
3) R→ SP		incentives abuse [7]	Low
4) **D →R**		**fare overcharging [66]**	**High**
5) D→ D		ride matching cheating [57]	Low
6) D→ SP		reputation cheating	Low
7) D→ SP		fees and incentives cheating [14]	Medium
8) **O→ R,D**		**account theft [8, 35]**	**High**

Table 4. Integrity threat taxonomy for RHSs (Risk-RH) based on four possible adversaries: riders (R), drivers (D), service provider (SP), and outsiders (O). The notation $X \rightarrow Y$ means X *attacks* Y.

B Threat Taxonomy Details

In Table 5, we present a more detailed description of the main threats against current RHSs introduced in Section 3.2 and the key reasons for their associated risk level.

Privacy Threats

Description	Risk	Risk Level Explanation
1) A malicious rider collects PII from drivers assigned to her.	Low	Medium impact: semi-scalable attack due to system design, limited reputation damage for the SP. Low likelihood: only few riders might attempt such attacks due to the reward, few reported incidents [2]
2) A malicious driver collects PII from riders she is assigned to.	Low	Low impact: non-scalable attack due to system design, limited reputation damage for the SP. Low likelihood: only few drivers might attempt such attacks due to the reward, no reported incidents.
3) A curious SP is able to track, sometimes in real-time, riders' precise location during their rides and infer private information from the data observed.	High	High impact: large scale inference attacks, significant reputation damage for the SP. High likelihood: ride data is collected by the SP by design, significant financial gain, reported incidents of misuse of this data [27, 31, 49].
4) A curious SP is able to track, sometimes in real-time, drivers' precise location while on service and infer private information from the data observed.	Medium	Medium impact: large scale inference attacks, drivers' ride data reveal less private information, reputation damage for the SP. Medium likelihood: ride data is collected by the SP by design, privacy-inference analysis is more difficult, less reward than riders data.
5) A malicious outsider can collect riders' PII in large scale by exploiting weaknesses on the services offered by the SP.	Medium	High impact: large scale attack, significant reputation damage to SP. Low likelihood: considerable reward, difficult to exploit because SP's services exposed little rider information, requires higher attack skills, no reported incidents.
6) A malicious outsider can collect drivers' PII in large scale by exploiting weaknesses on the services offered by the SP.	High	High impact: scalable due to system design, possible privacy violation of thousands of individuals, harvested data could be used to physically attack drivers. High likelihood: valuable information for some adversaries, system design facilitates the attack, current security mechanisms provided limited protection, reported related incidents [42, 60].
7) Malicious outsider gains (partial or total) unauthorized access to the SP's rider and/or driver registration databases.	High	High impact: significant reputation damage to the SP, possible privacy violation for millions of individuals. High likelihood: significant reward for the adversary, very common attack against online services, reported incidents [3].

Integrity Threats

Description	Risk	Risk Explanation
1) A malicious rider exploits a weakness on the system to pay lower fares.	Low	Low impact: non-scalable, limited financial loss, affects a small number of drivers. Medium likelihood: riders have limited input on fare calculation operations, limited financial gain, reported incidents [39], attacks are easy to detect.
2) A malicious rider exploits weaknesses on the system to improperly increase her reputation (e.g., to hide misbehavior).	Low	Low impact: non-scalable, limited reputation and financial cost to the SP. Low likelihood: rider reputation does not have high value, riders have little control over reputation operations, it may be easier to create a new rider account.
3) A malicious rider improperly accumulate incentives offered by the system for financial gain.	Low	Low impact: limited financial and reputation cost to the SP, incentives are location-dependent, SP can invalidate incentives. Medium likelihood: few incidents reported [7], interesting rewards for riders, abuse is easy to detect .
4) A malicious driver exploits weaknesses on the system to improperly charge riders a higher fare and/or fees.	High	High impact: large-scale attack if tools are made available, considerable reputation damage for the SP. Medium likelihood: drivers provide the inputs for fare calculation, smartphones are relatively easy to tamper with, direct financial gain, some related incidents has been reported [57, 66].
5) A malicious driver fakes her location or availability to increase her chances of being assigned to a ride.	Low	Low impact: limited scalability (only useful in some scenarios), limited financial impact to other drivers. Medium likelihood: attack is easy to execute, available tools, direct financial reward, reported incidents [7].
6) A malicious driver exploits weaknesses in the system to improperly increase her reputation.	Low	Medium impact: scalable if tools are made available, considerable reputation and financial loses for SP (e.g., , bad drivers are not banned). Low likelihood: difficult to exploit as drivers have read-only access to reputation information, significant motivation, no reported attacks.
7) A malicious driver exploits weaknesses on the system to improperly avoid fees or obtain incentives from the SP.	Medium	Medium impact: scalable if tools are made available, significant financial loses for the SP, incentives are location-dependent. Medium likelihood: significant financial motivation, reported incidents [14], some attacks are easy to detect.
8) A malicious outsider steals riders' and drivers' accounts by attacking the SP drivers or riders.	High	High impact: scalable attack, significant financial impact to riders and SP, significant reputation damage to the SP. High likelihood: common attack against online services, reported incidents [8], considerable reward.

Table 5. Extended description and explanation of risk level for the main threats of RHSs presented in Table 1 (Section 3.2).