Trust but Verify: Cryptographic Data Privacy for Mobility Management

Matthew Tsao, *Student, IEEE*, Kaidi Yang, Stephen Zoepf, and Marco Pavone, *Member, IEEE*

***Abstract*— The era of Big Data has brought with it a richer understanding of user behavior through massive data sets, which can help organizations optimize the quality of their services. In the context of transportation research, mobility data can provide Municipal Authorities (MA) with insights on how to operate, regulate, or improve the transportation network. Mobility data, however, may contain sensitive information about end users and trade secrets of Mobility Providers (MP). Due to this data privacy concern, MPs may be reluctant to contribute their datasets to MA. Using ideas from cryptography, we propose an interactive protocol between a MA and a MP in which MA obtains insights from mobility data without MP having to reveal its trade secrets or sensitive data of its users. This is accomplished in two steps: a commitment step, and a computation step. In the first step, Merkle commitments and aggregated traffic measurements are used to generate a cryptographic commitment. In the second step, MP extracts insights from the data and sends them to MA. Using the commitment and zero-knowledge proofs, MA can certify that the information received from MP is accurate, without needing to directly inspect the mobility data. We also present a differentially private version of the protocol that is suitable for the large query regime. The protocol is verifiable for both MA and MP in the sense that dishonesty from one party can be detected by the other. The protocol can be readily extended to the more general setting with multiple MPs via secure multi-party computation.**

***Index Terms*— Security and Privacy, Transportation Networks, Cyber-Physical systems, Networked Control Systems**

I. Introduction

The rise of mobility as a service, smart vehicles and smart cities is revolutionizing transportation industries all over the world. Mobility management, which entails operation, regulation, and innovation of transportation systems, can leverage mobility data to improve the efficiency, safety, accessibility, and adaptability of transportation systems far beyond what was previously achievable. The analysis and sharing of mobility data, however, introduces two key concerns. The first concern is data privacy; sharing mobility data can introduce privacy risks to end users that comprise the datasets. The second concern is credibility; in situations where data is not shared, how can the correctness of numerical studies be verified? These concerns motivate the need for data analysis tools for transportation systems which are both *privacy preserving* and *verifiable*.

The *data privacy* issue in transportation is a consequence of the trade-off between data availability and data privacy. While user data can be used to inform infrastructure improvement, equity and green initiatives, the data may contain sensitive user information and trade secrets of mobility providers. As a result, end users and mobility providers may be reluctant to share their data with city authorities. Cities have recently begun mandating micromobility providers to share detailed trajectory data of all trips, arguing that the data is needed to enforce equity or environmental objectives. Some mobility providers argued that while names and other directly identifiable information may not be included in the data, trajectory data can still reveal schedules, routines and habits of the city's inhabitants. The mobility providers' concern over the release of anonymized data is justified. [1] showed that any attempt to release anonymized data either fails to provide anonymity, or there are low-sensitivity attributes of the original dataset that cannot be determined from the published version. In general, anonymization is increasingly easily defeated by the very techniques that are being developed for many legitimate applications of big data [2]. Such disputes highlight the need for privacy-preserving data analysis tools in transportation.

A communication scheme between a sender and a receiver is *verifiable* if it enables the receiver to determine whether the message or report it receives is an accurate representation of the truth. When the objectives of mobility providers and policy makers are not aligned, one party may benefit from misreporting data or other information, giving rise to verifiability issues in transportation. An example of this is Greyball software [3]. Mobility providers developed Greyball software to deny service or display misleading information to targeted users. It was originally developed to protect their drivers from oppressive authorities in foreign countries, by misreporting driver location to accounts that were believed to belong to the oppressive authorities. However, mobility providers also used Greyball to hide their activity from authorities in the United States when their operations were scrutinized. Another example of verifiability issues is third party wage calculation apps [4]. Drivers, frustrated by instances of being underpaid, created an app to confirm whether the pay was consistent with the length and duration of each trip. Such incidents highlight

This research was supported by the National Science Foundation under CAREER Award CMMI-1454737 and Award CNS-1837135. K. Yang would like to acknowledge the support of the Swiss National Science Foundation (SNSF) Postdoc Mobility Fellowship (P400P2_199332).

M. Tsao, K. Yang and Marco Pavone are with Stanford University, Palo Alto, CA 94305 USA (e-mail: {mwtsao, ykd07, pavone}@stanford.edu).

S. Zoepf is with Lacuna Technologies, Palo Alto, CA 94306 (e-mail: stephen.zoepf@lacuna.ai).

the need for verifiable data analysis tools in transportation.

A. Statement of Contributions

In this paper we propose a protocol between a Municipal Authority and a Mobility Provider that enables the Mobility Provider to send insights from its data to the Municipal Authority in a privacy-preserving and verifiable manner. In contrast to *non-interactive* data sharing mechanisms (which are currently used by most municipalities) where a Municipal Authority is provided an aggregated and anonymized version of the data to analyze, our proposed protocol is an *interactive* mechanism where a Municipal Authority sends queries and Mobility Providers give responses. By sharing responses to queries rather than the entire dataset, interactive mechanisms circumvent the data anonymization challenges faced by non-interactive approaches [1], [2].

Our proposed protocol, depicted in Figure 1, has three main steps. In the first step, the Mobility Provider uses its data to produce a data identifier which it sends to the Municipal Authority. The Municipal Authority can then send its data query to the Mobility Provider in the second step. In the third step, the Mobility Provider sends its response along with a zero knowledge proof. The Municipal Authority can use the zero knowledge proof to check that the response is consistent with the identifier, i.e., the response was computed from the same data that was used to create the identifier. If the Municipal Authority has multiple queries, steps 2 and 3 are repeated.

The protocol uses cryptographic commitments and aggregated traffic measurements to ensure that the identifier is properly computed from the true mobility data. In particular, any deviation from the protocol by one party can be detected by the other, making the protocol strategyproof for both parties. Given that the identifier is properly computed, the zero knowledge proof then enables the Municipal Authority to verify the correctness of the response without needing to directly inspect the mobility data. Since the Municipal Authority never needs to inspect the mobility data, the protocol is privacy-preserving.

The protocol can be extended to the more general case of multiple Mobility Providers, each with a piece of the total mobility data. This is done by including a secure multi-party computation in step 3 of the protocol. Answering a large number of queries with our protocol can lead to privacy issues since it was shown in [5] that a dataset can be reconstructed from many accurate statistical measurements. To address this concern, we generalize the protocol to enable differentially private responses from the Mobility Provider in large query regimes.

B. Organization

This paper is organized as follows. The remainder of the introduction discusses academic work related to privacy and verifiability in transportation networks. In Section II we introduce a mathematical model of transportation networks and use it to formulate the data privacy problem for Mobility Management. We provide a high level intuitive description of our proposed protocol in Section III. In Section IV we provide

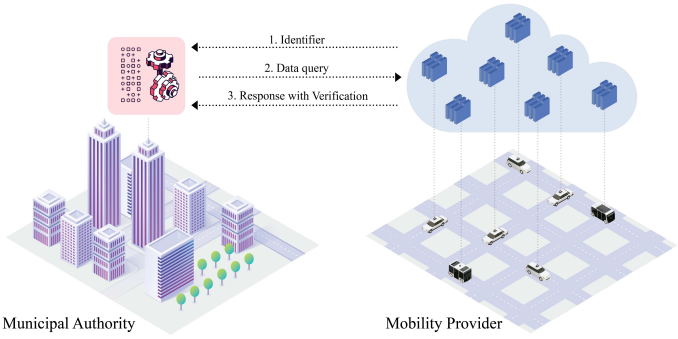

Fig. 1. The Mobility Provider can answer the Municipal Authority's data-related mobility queries in a verifiable way *without needing to share the data*. The absence of data sharing in the protocol reduces the chance that a malicious third party intercepts and uses the data for nefarious privacy-invasive purposes.

a full technical description of our protocol. We discuss some of the technical nuances of the protocol and their implications in Section V. We summarize our work and identify important areas for future research in Section VI. In Appendix I we present a differentially private extension of the protocol that is suitable for the large query regime.

C. Related Work

Within the academic literature, this work is related to the following four fields: misbehavior detection in cooperative intelligent transportation networks, data privacy in transportation systems, differential privacy, and secure multi-party computation. We briefly discuss how this work complements ideas from these fields.

Cooperative intelligent transportation networks (cITS) aim to provide benefits to the safety, efficiency, and adaptability of transportation networks by having individual vehicles share their information. As with all decentralized systems, security and robustness against malicious agents is essential for practical deployment. As such, misbehavior detection in cITS have been studied extensively [6]. Misbehavior detection techniques often rely on honest agents acting as referees, and are able to detect misbehavior in the honest majority setting. Watchdog is one such protocol [7], [8] which uses peer-to-peer refereeing. The protocol uses a public key infrastructure (PKI) to assign a persisting identity to each node in the network, and derives a reputation for each node based on its historical behavior. Our objective in this work is also detection of misbehavior, but in a different setting. In our setting, while the mobility network is comprised of many agents (customers and drivers), there is a single entity (the Mobility Provider, e.g., a ridehailing service) who is responsible for the storage and analysis of trip data. As such, the concept of honest majority does not apply to our setting. Furthermore, [8] does not address the issue of data privacy; indeed, PKIs can often expose the users' identities, especially if an attacker cross-references the network traffic with other traffic records.

Privacy in intelligent transportation systems is often implemented by using non-interactive anonymization (e.g., data aggregation), cryptographic tools or differential privacy. Providing anonymity in non-interactive data analysis mechanisms

is challenging [1], [2] and thus data aggregation alone is often not enough to provide privacy. From the cryptography side, to address the lack of anonymity provided by blockchains like Bitcoin and Ethereum, zero knowledge proofs [9] were deployed in blockchains like Zcash [10] to provide fully confidential transactions. In the context of transportation, zero knowledge proofs have been proposed for privacy-preserving vehicle authentication to EV charging services [11], and privacy-preserving driver authentication to customers in ride-hailing applications [12]. These privacy-preserving authentication systems rely on a trusted third party to distribute and manage certificates.

Differential privacy is an interactive mechanism for data privacy which uses randomized responses to hide user-specific information [1]. For any query, the data collector provides a randomized response, where two datasets which differ in only one entry produce statistically indistinguishable outputs. Due to this randomization, there is a trade-off between the accuracy of the response and the level of privacy provided. Randomization is necessary to preserve privacy in the large query regime as demonstrated by [5] which showed that a dataset can be reconstructed from many accurate statistical measurements. The standard model of differential privacy, however, relies on a *trusted* data collector to apply the appropriate randomized response to queries. This is problematic in situations where the data collector is not trusted. A local model of differential privacy where users perturb their data before sending it to the data collector has received significant attention due to trust concerns [13]. However mobility providers often record exact details about user trips, making local differential privacy unsuitable for current mobility applications (See Remark 14). Instead, we believe cryptographic techniques can be used to address trust concerns. There are also more general concerns about trust; downstream applications of data queries can lead to conflicts of interest and encourage strategic behavior.

Secure Multi-Party Computation (MPC) is a technique whereby several players, each possessing private data, can jointly compute a function on their collective data without any player having to reveal their data to other players [14]. MPC achieves confidentiality by applying Shamir's Secret Sharing [15] to inputs and intermediate results. In its base form, MPC is secure against honest-but-curious adversaries, which follow the protocol, but may try to do additional calculations to learn the private data of other players. In general, security against active malicious adversaries, which deviate from the protocol arbitrarily, requires a trusted third party to perform verified secret sharing [16]. In verified secret sharing, the trusted third party creates initial cryptographic commitments for each player's private data. The commitments do not leak any information about the data, and allows honest players to detect misbehavior using zero knowledge proofs. MPC is a very promising tool for our problem, but a trusted third party able to eliminate strategic behavior does not yet exist in the transportation industry, therefore a key objective of this work is to develop mechanisms to defend against strategic behavior.

In Summary - Our goal in this work is to develop a protocol that enables a mobility provider to share insights from its data to a municipal authority in a privacy-preserving and verifiable manner. Existing work in accountability and misbehavior detection focus on networks with many agents and rely on honest majority. Such assumptions, however, are not realistic for interactions between a municipal authority and a few mobility providers. We thus turn our attention to differential privacy and secure multi-party computation which provide data privacy but require honesty of participating parties. To address this, we develop mechanisms based on cryptography and aggregated roadside measurements to detect dishonest behavior.

II. Model & Problem Description

In this section we present a model for a city's transportation network and formulate a data Privacy for Mobility Management (PMM) problem. Section II-A introduces a mathematical representation of a city's transportation network along with the demand and mobility providers. In Section II-B we formalize the notion of data privacy using secure multi-party computation, and introduce assumptions on user behavior that we will need to construct verifiable protocols. We then formally introduce the PMM problem and describe several transportation problems that can be formulated in the PMM framework.

A. Transportation Network Model

Transportation Network - Consider the transportation network of a city, which we represent as a directed graph $G = (V, E, f)$ where vertices represent street intersections and edges represent roads. For each road $e \in E$ we use an increasing differentiable convex function $f_e : \mathbb{R}_+ \to \mathbb{R}_+$ to denote the travel cost (which may depend on travel time, distance, and emissions). of the road as a function of the number of vehicles on the road. We will use $n := |V|$ and $m := |E|$ to denote the total number of vertices and edges in G respectively. Time is represented in discrete timesteps of size Δt. The operation horizon is comprised of $T+1$ timesteps as $\mathcal{T} := \{0, \Delta t, 2\Delta t, ..., T\Delta t\}$.

Mobility Provider - A Mobility Provider (MP) is responsible for serving the transportation demand. It does so by choosing a routing x of its vehicles within the transportation network. The routing must satisfy multi-commodity network flow constraints (see Supplementary Material B.1 and B.2 of the extended version [17] for explicit descriptions of these constraints) and the MP will choose a feasible flow that maximizes its utility function J_{MP}. Some examples of MPs are ridehailing companies, bus companies, train companies, and micromobility (i.e., bikes & scooters) companies.

Transportation Demand Data - The MP's demand data is a list of completed trips $\Lambda := \{\lambda_1, ..., \lambda_q\}$, where λ_i contains the following basic metadata about the ith trip: Pickup and dropoff locations, request time, match time (i.e., the time at which the user is matched to a driver), pickup and dropoff time, driver wage, trip fare, trip trajectory (i.e., the vehicle's trajectory from the time the vehicle is matched to the rider until the time the rider is dropped off at their destination) and properties of the service vehicle.

For locations $i, j \in V$ and a timestep t, we use $\Lambda(i, j, t)$ to denote the number of users in the data set who request transit from location i to location j at time t.

Remark 1 (Multiple Mobility Providers)**.** We can consider settings where there are multiple mobility providers, $\text{MP}_1, \text{MP}_2, ..., \text{MP}_\ell$, where Λ_j is the demand data of MP_j. The demand data set for the whole city is thus $\Lambda = \cup_{j=1}^{\ell} \Lambda_j$.

Ridehailing Periods - For MPs that operate ridehailing services, a ridehailing vehicle's trajectory is often divided into three different periods (with Period 0 often ignored):

Period 0: The vehicle is not online with a platform. The driver may be using the vehicle personally.

Period 1: The vehicle is vacant and has not yet been assigned to a rider.

Period 2: The vehicle is vacant, but it has been assigned to a rider, and is en route to pickup.

Period 3: The vehicle is driving a rider from its pickup location to its dropoff location.

B. Objective: Privacy for Mobility Management (PMM)

In the data Privacy for Mobility Management (PMM) problem, a Municipal Authority (MA) wants to compute a function $g(\Lambda)$ on the travel demand, where $g(\Lambda)$ is some property of Λ that can inform MA on how to improve public policies. There are two main obstacles to address: privacy and verifiability.

Privacy issues arise since trip information may contain sensitive customer information as well as trade secrets of Mobility Providers (MP). For this reason MPs may be reluctant to contribute their data for MA's computation of $g(\Lambda)$. This motivates the following notion of privacy:

Definition 1 (Privacy in Multi-Party Computation)**.** Suppose $\text{MP}_1, ...\text{MP}_\ell$ serve the demands $\Lambda_1, ..., \Lambda_\ell$ respectively, and we denote $\Lambda = \cup_{i=1}^{\ell} \Lambda_i$. We say a protocol for computing $g(\Lambda)$ between a MA and several MPs is privacy preserving if

1) MA learns nothing about Λ beyond the value of $g(\Lambda)$.
2) For any pair $i \neq j$, MP_i learns nothing about Λ_j beyond the value of $g(\Lambda)$.

Verifiability issues arise if there is incentive misalignment between the players. In particular, if the MA or a MP can increase their utility by deviating from the protocol, then the computation of $g(\Lambda)$ may be inaccurate. To address this issue, we need the protocol to be verifiable, as described by Definition 2. The following assumption is necessary to ensure accurate reporting of demand (See Supplementary Material B.5 of the extended version [17] for more details):

Assumption 1 (Strategic Behavior)**.** *We assume in this work that drivers and customers of the transportation network will behave honestly (by this we mean they will always follow the protocol), but MA and MPs may act strategically to maximize their own utility functions.*

Definition 2 (Verifiable Protocol)**.** A protocol for computing $g(\Lambda)$ is verifiable under Assumption 1 if:

1) Any deviation from the protocol by the MA can be detected by the MPs provided that all riders and drivers act honestly (i.e., follow the protocol).
2) Any deviation from the protocol by an MP can be detected by the MA provided that all riders and drivers act honestly.

Our objective in this paper is to present a PMM protocol, which is defined below.

Definition 3 (PMM Protocol)**.** A PMM protocol between a MA and $\text{MP}_1, ...\text{MP}_\ell$ can, given any function g, compute $g(\Lambda)$ for MA while ensuring privacy and verifiability as described by Definitions 1 and 2 respectively.

Remark 2 (Admissible Queries and Differential Privacy)**.** While a PMM protocol hides all information about Λ beyond the value of $g(\Lambda)$, $g(\Lambda)$ itself may contain sensitive information about Λ. The extreme case would be if g is the identity function, i.e., $g(\Lambda) = \Lambda$. In such a case, the MPs should reject the request to protect the privacy of its customers. More generally, MPs should reject functions g if $g(\Lambda)$ is highly correlated with sensitive information in Λ. The precise details as to which functions g are deemed acceptable queries must be decided upon beforehand by MA and the MPs together.

Differential privacy mechanisms provide a principled way to address the sensitivity of g by having MPs include noise in the computation of $g(\Lambda)$. If the noise distribution is chosen according to both the desired privacy level and the sensitivity of g to its inputs, then the output is differentially private. Note that this privacy is not for free; the noise reduces the accuracy of the output. The precise choice of noise distribution is important for both the privacy and accuracy of this method, so ensuring that the randomization step is conducted properly in the face of strategic MAs and MPs is essential. This can be done with a combination of coinflipping protocols and secure multi-party computation, which we describe in Appendix I.

We now present some important social decision making problems that can be formulated within the PMM framework. Note that these applications are offline decision making problems and thus do not impose strict requirements on computation times of protocols. Regulation checks can be conducted daily or weekly, and infrastructure improvement initiatives are seldom more frequent than one per week. The low frequency of such queries gives plenty of time to compute a solution. For this reason, we do not expect the computational complexity of the solution to be an issue.

1) Regulation Compliance for Mobility Providers: Suppose MA wants to check whether a MP is operating within a set of regulations $\rho_1, ..., \rho_k$. The metadata contained within each trip includes request time, match time, pickup time, dropoff time, and trip trajectory, which can be used to check regulation compliance. If we define the function $\rho_i(\Lambda)$ to be 1 if and only if regulation i is satisfied, and 0 otherwise, then regulation compliance can be determined from the function $g(\Lambda) := \prod_{t=1}^{k} \rho_t(\Lambda)$. Below are some examples of regulations that can be enforced using trip metadata.

Example 1 (Waiting Time Equity)**.** MP is not discriminating against certain requests due to the pickup or droppoff locations. Specifically, the difference in average waiting time among different regions should not exceed a specified regula-

tory threshold.

Example 2 (Congestion Contribution Limit)**.** The contribution of MP vehicles (in Period 2 or 3) to congestion should not exceed a specified regulatory threshold.

Example 3 (Accurate Reporting of Period 2 Miles)**.** A ride-hailing driver's pay per mile/minute depends on which period they are in. In particular, the earning rate for period 2 is often greater than that of period 1. For this reason, mobility providers are incentivized to report period 2 activity as period 1 activity. To protect ridehailing drivers, accurate reporting of period 2 activity should be enforced.

Example 4 (Emissions Limit)**.** The collective emission rate of MP vehicles in Phases 2 and 3. should not exceed a specified regulatory threshold. MP emissions can be computed from the metadata of served trips, in particular the trajectory and vehicle make and model.

See Supplementary Material B.4 of the extended version [17] for further details on formulating the above examples within the PMM framework.

2) Transportation Infrastructure Development Projects: *Transportation Infrastructure Improvment Projects* - A Municipal Authority (MA) measures the efficiency of the current transportation network via a concave social welfare function $J_{\text{MA}}(x)$. The MA wants to make improvements to the network G through infrastructure improvement projects. Below are some examples of such projects.

Example 5 (Building new roads)**.** The MA builds new roads E_{new} so the set of roads is now $E \cup E_{\text{new}}$, i.e., G now has more edges.

Example 6 (Building Train tracks)**.** The MA builds new train routes. Train routes differ from roads in that the travel time is independent of the number of passengers, i.e., there is no congestion effect.

Example 7 (Adding lanes to existing roads)**.** The MA adds more lanes to some roads $E' \subset E$. As a consequence, the shape of f_e will change for each $e \in E'$.

Example 8 (Adjusting Speed limits)**.** Similar to adding more lanes, adjusting the speed limit of a road will change its delay function.

Evaluation of Projects - We measure the utility of a project using a Social Optimization Problem (SOP). An infrastructure improvement project θ makes changes to the transit network, so let G_θ denote the transit network obtained by implementing θ. The routing problem ROUTE(θ, Λ) associated with θ is the optimal way to serve requests in G_θ as measured by MP's objective function J_{MP}. Letting $S_{\theta,\Lambda}$ be the set of flows satisfying multi-commodity network flow constraints (See Supplementary Material B.1 and B.2 of the extended version [17] for time-varying and steady state formulations respectively). for the graph G_θ and demand Λ, ROUTE(θ, Λ) is given by

$$\begin{aligned} &\max\ J_{\text{MP}}(x) \\ &\text{s.t. } x \in S_{\theta,\Lambda}. \end{aligned} \tag{ROUTE(θ,Λ)}$$

Definition 4 (The Infrastructure Development Selection Problem)**.** Suppose there are k infrastructure improvement projects $\Theta := \{\theta_1, \theta_2, ..., \theta_k\}$ available, but the city only has the budget for one project. The city will want to implement the project that yields the most utility, which is determined by the following optimization problem.

$$\underset{1\le i\le k}{\text{argmax}}\ J_{\text{MA}}\left(\underset{x\in S_{\theta_i,\Lambda}}{\text{argmax}}\ J_{\text{MP}}(x)\right). \tag{SOP(Θ,Λ)}$$

In the context of PMM, the function g associated with the infrastructure development selection problem is $g(\Lambda) := \text{SOP}(\Theta, \Lambda)$.

3) Congestion Pricing: Some ridehailing services allow drivers to choose the route they take when delivering customers. When individual drivers prioritize minimizing their own travel time and disregard the negative externalities they place on other travelers, the resulting user equilibrium can experience significantly more congestion than the social optimum. In these cases, the total travel time of the user equilibrium is larger than that of the social optimum. This gap, known as the price of anarchy, is well studied in the congestion games literature.

Congestion pricing addresses this issue by using road tolls to incentivize self-interested drivers to choose routes so that the total travel time of all users is minimized. The desired road tolls depend on the demand Λ, so MA would need help from MPs to compute the prices. Congestion pricing can be formulated in the PMM framework through the query function g_{cp} described in (2).

When the travel cost is the same as travel time, the prices can be obtained from the Traffic Assignment Problem [18]:

$$\begin{aligned} &\min_x \sum_{e\in E} x_e f_e(x_e) \\ &\text{s.t. } x = \sum_{o\in V}\sum_{d\in V} x^{od} \\ &\quad x^{od} \succeq 0\ \forall o \in V, d \in V \\ &\quad \sum_{(u,v)\in E} x^{od}_{(u,v)} - x^{od}_{(v,u)} = \Lambda(o,d)\left(\mathbb{1}_{[u=o]} - \mathbb{1}_{[u=d]}\right)\forall u \in V \end{aligned} \tag{1}$$

where x_e^{od} denotes the traffic flow from o to d that uses edge e. The objective measures the sum of the travel times of all requests in Λ. The desired prices are then given by:

$$g_{\text{cp}}(\Lambda) := \{x_e^* f_e'(x_e^*)\}_{e\in E} \text{ where } x^* \text{ solves (1).} \tag{2}$$

See Supplementary Material B.8 of the extended version [17] for more details on congestion pricing.

III. A HIGH LEVEL DESCRIPTION OF THE PROTOCOL

We focus our discussion on the case where there is one MP. The protocol we will present can be generalized to the

multiple MP setting through secure Multi-party Computation [14].

In this paper we present a verifiable interactive protocol, which allows MA to check whether or not the message it receives from MP is in fact $g(\Lambda)$. This will result in a protocol where MA is able to obtain $g(\Lambda)$ without requiring MP to reveal any information about Λ beyond the value of $g(\Lambda)$.

First, we describe a non-confidential way to compute $g(\Lambda)$. We will discuss how to make it confidential in the next paragraph. MP will send a commitment $\sigma = \mathsf{MCommit}(\Lambda, r)$ of Λ to MA. This commitment will enable MA to certify that the result given to it by MP is computed using the true demand Λ. The commitment is confidential, meaning it reveals nothing about Λ, and is binding, meaning that it will be inconsistent with any other demand $\Lambda' \neq \Lambda$. Now suppose MP computes a message $z = g(\Lambda)$. To convince MA that the calculation is correct, MP will construct a witness $w := (\Lambda, r)$. When MA receives the message z and witness w, it will compute $C(\sigma, z, w)$, where C is an evaluation algorithm. $C(\sigma, z, w)$ evaluates to `True` if

1) Rider Witness and Aggregated Roadside Audit checks are satisfied. (σ was reported honestly)
2) $\mathsf{MCommit}(\Lambda, r) = \sigma$. ($\Lambda$ is the demand that was used to compute σ).
3) $g(\Lambda) = z$ (g was evaluated properly.)

If any of these conditions are not met, $C(\sigma, z, w)$ will evaluate to `False`. Finally, MA will accept the message z only if $C(\sigma, z, w) = \texttt{True}$.

The approach presented in the previous paragraph is not privacy-preserving because the witness w being sent from MP to MA includes the demand Λ. Fortunately, we can use zero knowledge proofs to obtain privacy. Given an arithmetic circuit C (which in our case is the evaluation algorithm C), it is possible for one entity (the prover) to convince another entity (the verifier) that it knows an input z, w so that $C(\sigma, z, w) = \texttt{True}$ without revealing what w is. This is done by constructing a zero knowledge proof π from (z, w) and sending (z, π) to the verifier instead of sending (z, w). MA can then check whether π is a valid proof for z. The proof π is zero knowledge in the sense that it is computationally intractable to deduce anything about w from π, aside from the fact $C(\sigma, z, w) = \texttt{True}$. For our application, the prover will be MP who is trying to convince the verifier, which is MA, that it computed $g(\Lambda)$ correctly.

This protocol requires MP to send a commitment of the true demand data to MA. This is problematic if MP has incentive to be dishonest, i.e., provide a commitment corresponding to a different dataset. To ensure this does not happen, our protocol uses a Rider Witness incentive to prevent MP from underreporting demand, and Aggregated Roadside Audits to prevent MP from overreporting demand. These two mechanisms establish the verifiability of the protocol, since, as seen in first requirement of C, MA will reject the message if either of these mechanisms detect dishonesty.

In Summary - We present a verifiable interactive protocol. First, MP sends a commitment of the demand to MA, which ensures that the report is computed using the true demand. The correctness of this commitment is enforced by Rider Witness and Aggregated Roadside Audits. MA then announces the function g that it wants to evaluate. MP computes a message $z \leftarrow g(\Lambda)$ and constructs a witness w to the correctness of z. Since w in general contains sensitive information, it cannot be used directly to convince MA to accept the message z. MP computes a zero knowledge proof π of the correctness of z from w, and sends the message z and proof π to MA. MA accepts z if π is a valid zero knowledge proof for z.

Implementation - To implement our protocol we will use several tools from cryptography. The commitment σ is implemented as a Merkle commitment. For computing zero knowledge proofs, we will need a zk-SNARK that doesn't require a trusted setup. PLONK [19], Sonic [20], and Marlin [21] using a DARK based polynomial commitment schemes described in [22], [23]. Other options include Bulletproofs [24] and Spartan [25]. The cryptographic tools used in the protocol are reviewed in Supplementary Material B.3 of the extended version [17].

IV. The Protocol

In this section we present our protocol for the PMM problem described in Section II-B. For clarity and simplicity of exposition we will focus on the case where there is one Mobility Provider. The single MP case can be extended to the multiple MP case via secure multi-party computation [14]. We present the protocol, which is illustrated in Figure 2, in Section IV-A. In Section IV-B we discuss mechanisms used to ensure verifiability of the protocol.

The protocol uses the following cryptographic primitives: hash functions, commitment schemes, Merkle trees, public key encryption and zero knowledge proofs. Hash functions map data of arbitrary size to fixed size messages, often used to provide succinct identifiers for large datasets. Commitment schemes are a form of verifiable data sharing where a receiver can reserve data from a sender, obtain the data at a later point, and verify that the data was not changed between the reservation and reception times. A Merkle tree is a particular commitment scheme we will use. In public key encryption, every member of a communication network is endowed with a public key and a private key. The public key is like a mailbox which tells senders how to reach the member, and the secret key is the key to the mailbox, so messages can be viewed only by their intended recipients. Zero knowledge proofs, as discussed in Section III, enable a prover to convince a verifier that it knows a solution to a mathematical puzzle without directly revealing its solution. For a more detailed description of these concepts, we refer the reader to the Supplementary Material B.3 of the extended version [17], where we provide a self-contained introduction of the cryptographic tools used in this work.

A. Protocol Description

The protocol entails 6 stages:

Stage 0: (Data Collection) MP serves the demand Λ and builds a Merkle Tree T_Λ of the demand it serves. MP publishes the root of T_Λ, which is denoted as $\sigma := \mathsf{MCommit}(\Lambda, r)$ so

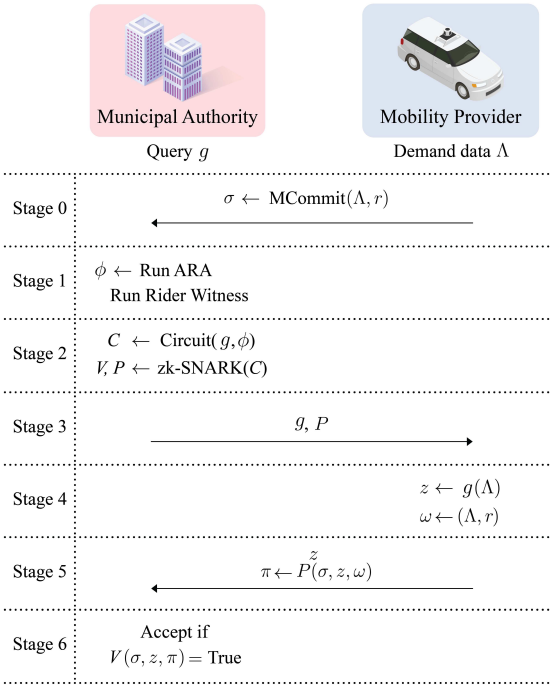

Fig. 2. A block diagram of the communication between MA and MP.

that MA, all riders and all drivers have access to σ. Here r is the set of nonces used to make the commitment confidential.

Stage 1: (Integrity Checks) MA instantiates Rider Witness and Aggregated Roadside Audits to ensure that σ was computed using the true demand Λ. The description of these mechanisms can be found in Section IV-B.

Stage 2: (Message Specifications) MA specifies to MP the function g it wants to compute.

Stage 3: (zk-SNARK Construction) MA constructs an evaluation algorithm C for the function g. σ, z are public parameters of C, and the input to C is a witness of the form $w = (\Lambda_w, r_w, c_w)$, where r_w is a set of nonces, Λ_w is a demand matrix, and c_w is an optional input that may depend on g (See Remark 4). C does the following:

1) Checks whether the Rider Witness and Aggregated Roadside Audit tests are satisfied (This checks that σ was reported honestly),
2) Checks whether $\mathsf{MCommit}(\Lambda_w, r_w) = \sigma$ (This determines whether the provided demand Λ_w is the same as the demand that created σ),
3) Checks whether $g(\Lambda_w) = z$ (This checks that the message z is computed properly from Λ_w).

C will evaluate to `True` if and only if all of those checks pass. Now, using one of the schemes from [19]–[21], [24], [25], MA will create a zk-SNARK (S, V, P) for C. S is a set of public parameters that describes the circuit C, P is a prover function which MP will use to construct a proof, and V is a verification function which MA will use to verify the correctness the MP's proof. It sends $C, (S, V, P), g$ to MP.

Stage 4: (Function Evaluation) If the request g is not a privacy-invasive function (see Remark 2), MP will compute a message $z = g(\Lambda)$ and construct a witness $w := (\Lambda, r, c_w)$ to the correctness of z.

Stage 5: (Creating a Zero Knowledge Proof) MP uses the zk-SNARK's prover function P to construct a proof $\pi := P(\sigma, z, w)$ that certifies the calculation of z. MP sends z, π to MA.

Stage 6: (zk-SNARK Verification) MA uses the zk-SNARK's verification function $V(\sigma, z, \pi)$ to check whether MP is giving a properly computed message. If this is the case, MA accepts the message z.

Remark 3 (Computational Gains via Commit-then-Prove)**.** Steps 2) and 3) of the evaluation circuit C involve different types of computation. This heterogeneity can introduce computational overhead in the zk-SNARK. Commit-and-Prove zk-SNARKs [26], [27] are designed to handle computational heterogeneities, however existing implementations require a trusted setup.

Remark 4 (Verifying solutions to convex optimization problems)**.** If $g(\Lambda_w)$ is the solution to a convex optimization problem parameterized by Λ_w, (e.g., $g(\Lambda_w) = \mathrm{SOP}(\Theta, \Lambda_w)$ or congestion pricing $g_{\mathrm{cp}}(\Lambda_w)$), then computing $g(\Lambda_w)$ within the evaluation algorithm C may cause C to be a large circuit, thus making evaluation of C computationally expensive. Fortunately, this can be avoided by leveraging the structure of convex problems. If $z = g(\Lambda_w)$, we can include the optimal primal and dual variables associated with z in the optional input c_w. This way, checking the optimality of z can be done by checking that c_w satisfy the KKT conditions rather than needing to re-solve the problem.

B. Ensuring accuracy of σ

The protocol presented in the previous section requires MP to share a commitment to the true demand Λ. However, scenarios exist where the MP may face direct or indirect incentives to misreport demand, such as per-ride fees, congestion charges, or other regulations that may constrain MP operations. In this section we present mechanisms to ensure that MP submits a commitment $\sigma = \mathsf{MCommit}(\Lambda, r)$ corresponding to the true demand Λ rather than a commitment $\sigma' = \mathsf{MCommit}(\Lambda', r)$ corresponding to some other demand Λ'. Specifically, we present Rider Witness and Aggregated Roadside Audits which detect underreporting and overreporting of demand respectively.

1) Rider Witness: Detecting underreported demand: In this section, we present a Rider Witness mechanism to detect omission or tampering of the demand Λ. Concretely, if a MP sends to MA a Merkle commitment $\sigma' = \mathsf{MCommit}(\Lambda', r)$ which underreports demand, i.e., $\Lambda \setminus \Lambda'$ is non-empty, then Rider Witness will enable MA to detect this. MA can impose fines or other penalties when such detection occurs to deter MP from underreporting the demand.

Rider Witness Incentive Mechanism - At the beginning of Stage 0 (Data Collection) of the protocol, MP constructs a public key and private key pair $(\mathsf{pk}_{\mathrm{mp}}, \mathsf{sk}_{\mathrm{mp}})$ to use for digital signatures. The payment process is as follows: When the ith customer is delivered to their destination, the customer will send a random nonce r_i to MP. MP will respond with a receipt $(H(r_i||\lambda_i), \sigma_i)$, where $\sigma_i := \mathsf{sign}(\mathsf{sk}_{\mathrm{mp}}, H(r_i||\lambda_i))$ is a digital

signature certifying that MP recognizes λ_i as an official ride (here $||$ represents concatenation of binary strings). Here H is SHA256, so that $H(r_i||\lambda_i)$ is a cryptographic commitment to the trip λ_i. The customer is required to pay the trip fare only if $\mathsf{verify}(\mathsf{pk}_{\text{mp}}, H(r_i||\lambda_i), \sigma_i) = \texttt{True}$, i.e., they received a valid receipt.

Definition 5 (Rider Witness Test)**.** Given a commitment σ' reported by MP to MA, each rider who was served by MP requests a Merkle proof that their ride is included in the computation of σ'. If there exists a valid[1] ride receipt $(H(r_i||\lambda_i), \sigma_i)$ for which MP cannot provide a Merkle proof, then the customer associated with λ_i will report $(H(r_i||\lambda_i), \sigma_i)$ to MA. MA checks if σ_i is a valid signature for $H(r_i||\lambda_i)$, and if so, directly asks MP for a Merkle Proof that λ_i is included in the computation of σ'. If MP is unable to provide the proof, then σ' fails the Rider Witness Test.

Observation 1 (Efficacy of Rider Witness)**.** *Under Assumption 1, if MP submits a commitment $\sigma' = \mathit{MCommit}(\Lambda', r)$ which omits a ride, i.e., $\Lambda \setminus \Lambda'$ is non-empty, then σ' will fail the Rider Witness Test.*

Proof of Observation 1. If $\Lambda \not\subseteq \Lambda'$, then there exists some λ_i which is in Λ but not Λ'. Suppose Alice was the rider served by ride λ_i. Forging a proof that $\lambda_i \in \Lambda'$ requires finding a hash collision for the hash function used in the Merkle commitment. Since $\mathsf{MCommit}$ is implemented using a cryptographic hash function (e.g., SHA256), it is computationally intractable to find a hash collision, and thus MP will be unable to forge a valid proof that $\lambda_i \in \Lambda'$.

If MP does not provide Alice a valid proof within a reasonable amount of time (e.g., several hours), Alice can then report $(H(r_i||\lambda_i), \sigma_i)$ to MA. This reporting does not compromise Alice's privacy due to the hiding property of cryptographic hash functions. MA will check whether $\mathsf{verify}(\mathsf{pk}_{\text{mp}}, H(r_i||\lambda_i), \sigma_i) = \texttt{True}$, and if so, means that λ_i is recognized as a genuine trip by MP. MA will directly ask MP for a Merkle proof that $H(r_i||\lambda_i) \in T_\Lambda$. Since MP cannot provide a valid proof, this is evidence that a genuine trip was omitted in the computation of σ', and hence σ' will fail the Rider Witness test. □

Remark 5 (Tamperproof Property)**.** We note that Rider Witness also prevents the MP from altering the data associated with genuine rides. If MP makes changes to $\lambda_i \in \Lambda$ resulting in some λ_i', then by collision resistance of H, it is computationally infeasible to find r' so that $H(r_i||\lambda_i) = H(r_i'||\lambda_i')$. If such a change is made, then $H(r_i'||\lambda_i')$ is included into the computation of σ' instead of $H(r_i||\lambda_i)$. This means $(H(r_i||\lambda_i), \sigma_i)$ becomes a valid witness that data tampering has occurred.

Remark 6 (Receipts are Unforgeable)**.** Note that it is not possible for a rider to report a fake ride $\lambda' \notin \Lambda$ to MA. This is because the corresponding signature σ' cannot be forged without knowing MP's secret key sk_{mp}. Therefore, assuming sk_{mp} is only known to MP, only genuine trips can be reported.

[1]In the sense that $\mathsf{verify}(\mathsf{pk}_{\text{mp}}, H(r_i||\lambda_i), \sigma_i) = \texttt{True}$.

Remark 7 (Honesty of riders)**.** The Rider Witness mechanism assumes that riders are honest, i.e., they will not collude with MP by accepting invalid receipts.

2) Aggregated Roadside Audits: Detecting overreported demand: In this section we present an Aggregated Roadside Audit (ARA) mechanism to detect overreporting of demand. Concretely, if MP announces a commitment $\sigma' = \mathsf{MCommit}(\Lambda', r)$, where Λ' is a strict superset of Λ (i.e., $\Lambda' \setminus \Lambda$ is non-empty), then ARA will enable MA to detect this. Thus between ARA and Rider Witness, MA can detect if MP commits to a demand that is not Λ.

Aggregated Roadside Audits - Due to the Rider Witness mechanism, we can assume that MP submits a commitment σ' computed from Λ' satisfying $\Lambda \subseteq \Lambda'$, i.e., Λ' is a superset of Λ. For an edge $e \in E$ and a demand Λ, define

$$\varphi(e, \Lambda) := \sum_{\lambda \in \Lambda} \mathbb{1}_{[\lambda \text{ traverses } e]} \tag{3}$$

to be the number of trips that traversed e during passenger pickup (Period 2) or passenger delivery (Period 3). Since trip route is provided in the trip metadata, $\varphi(e, \Lambda)$ can be computed from Λ.

Definition 6 (ARA Test)**.** The Aggregated Roadside Audit places a sensor on every road to conduct an audit on each road $e \in E$ to measure $\varphi(e, \Lambda)$. These values are then aggregated as $\phi := \sum_{e \in E} \varphi(e, \Lambda)$. A witness $w = (\Lambda_w, r_w, c_w)$ passes the ARA test if and only if

$$\sum_{e \in E} \varphi(e, \Lambda_w) = \phi. \tag{ARA}$$

Observation 2 (Efficacy of Aggregated Roadside Audits)**.** *Under Assumption 1, if MP submits a commitment $\sigma' = \mathit{MCommit}(\Lambda', r)$ to a strict superset of the demand, i.e., $\Lambda \subset \Lambda'$, then any proof submitted by MP will either be inconsistent with σ' or will fail the ARA test. Hence MP cannot overreport demand.*

Proof of Observation 2. Suppose Λ' is a strict superset of Λ, which means that there exists some $\lambda' \in \Lambda' \setminus \Lambda$. Then there must exist some $e' \in E$ for which $\varphi(e', \Lambda') > \varphi(e', \Lambda)$. In particular, any edge in the trip route of λ' will satisfy this condition. With the inclusion of the ARA test, MP is unable to provide a valid witness for MA's evaluation algorithm C (and as a consequence, will be unable to produce a valid zero knowledge proof) for the following reason:

1) $\mathsf{MCommit}$ is a collision-resistant function (since it is built using a cryptographic hash function H), so because $\sigma' = \mathsf{MCommit}(\Lambda', r)$, it is computationally intractable for MP to find $\Lambda'' \neq \Lambda'$ and nonce values r'' so that $\mathsf{MCommit}(\Lambda'', r'') = \sigma'$. Therefore, in order to satisfy condition 2 of C (see *Stage 3* of Section IV-A), MP's witness must choose Λ_w to be Λ'.
2) However, Λ' will not pass the ARA test. To see this, note that (a) $\Lambda \subseteq \Lambda'$ implies that $\varphi(e, \Lambda) \leq \varphi(e, \Lambda')$ for all $e \in E$. Furthermore, (b) there exists an edge e' where the inequality is strict, i.e., $\varphi(e', \Lambda) < \varphi(e', \Lambda')$.

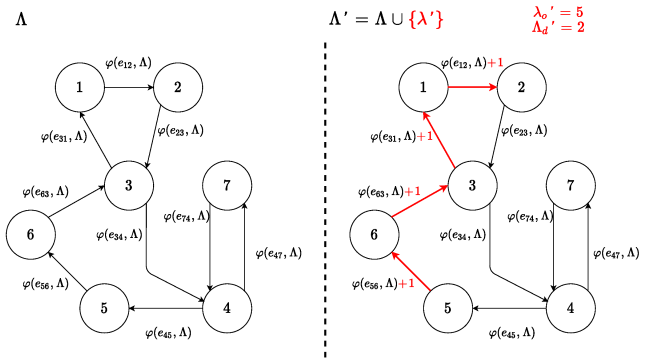

Fig. 3. An example of ARA. The true demand is $\boldsymbol{\Lambda}$, which results in traffic shown on the left. Here $\boldsymbol{\varphi(e_{ij}, \Lambda)}$ is the total number of trips in $\boldsymbol{\Lambda}$ that use the edge from i to j. Suppose MP submits a commitment to $\boldsymbol{\Lambda' = \Lambda \cup \{\lambda'\}}$, i.e., inserts a fake trip $\boldsymbol{\lambda'}$ into the commitment. In this example, $\boldsymbol{\lambda'}$ is a fake trip from $\mathbf{5}$ to $\mathbf{2}$ that MP claims was served via the route $\boldsymbol{\{e_{56}, e_{63}, e_{31}, e_{12}\}}$ (shown in red on the right). $\boldsymbol{\lambda'}$ increases the total traffic on the roads $\boldsymbol{e_{56}, e_{63}, e_{31}, e_{12}}$ and as a result, we have $\boldsymbol{\sum_{e\in E} \varphi(e, \Lambda') = \phi + 4}$.

From this, we see that

$$\begin{aligned}\phi = \sum_{e\in E} \varphi(e,\Lambda) &= \varphi(e',\Lambda) + \sum_{e\in\Lambda, e\neq e'} \varphi(e,\Lambda) \\ &\overset{(a)}{\leq} \varphi(e',\Lambda) + \sum_{e\in\Lambda, e\neq e'} \varphi(e,\Lambda') \\ &\overset{(b)}{<} \varphi(e',\Lambda') + \sum_{e\in\Lambda, e\neq e'} \varphi(e,\Lambda') \\ &= \sum_{e\in E} \varphi(e,\Lambda'),\end{aligned}$$

i.e., if the witness passes condition 2 of C, then it will fail the ARA test.

□

Therefore the value of ϕ can be used to detect fictitious rides. See Figure 3 for a visualization of ARA. In the following remark, we present a variant of ARA that is robust to measurement errors.

Remark 8 (Error Tolerance in ARA)**.** Trip trajectories are often recorded via GPS, so GPS errors can lead to inconsistencies between ARA sensor measurements and reported trajectories. To prevent an honest MP from failing the ARA test due to GPS errors, one can use an error tolerant version of the ARA test defined below

$$\left|\phi - \sum_{e\in E} \varphi(e, \Lambda_w)\right| \leq \epsilon\phi$$

where $\epsilon \in [0, 1]$ is a tuneable tolerance parameter to account for GPS errors while still detecting non-negligible overreporting of demand.

Remark 9 (Honesty of Drivers)**.** The correctness of ARA presented in Observation 2 assumes that drivers are honest when declaring their current period to ARA sensors, e.g., a driver who is in period 3 will not report themselves as period 1 or 2.

Two challenges that arise in the computation of ϕ are privacy and honesty, which are described below.

Remark 10 (Privacy-Preserving computation of ϕ)**.** The naïve way to compute ϕ is for MA to collect the values $\varphi(e, \Lambda)$ from each road. This, however, can compromise data privacy. Indeed, if there is only 1 request in Λ, then measuring the number of customer carrying vehicles that traverse each link exposes the trip route of that request: Edges that are traversed 1 time are in the route, and edges that are traversed 0 times are not. More generally, observing $\varphi(e, \Lambda)$ on all roads $e \in E$ exposes trip routes to or from very unpopular locations.

Remark 11 (Honest computation of ϕ)**.** It is essential that MA acts truthfully when taking measurement and computing ϕ in ARA, otherwise MP will be wrongfully accused of dishonesty.

Fortunately, the ARA sensors can use public key encryption to share their data with each other to compute ϕ in a privacy-preserving and honest way so that MA cannot learn $\varphi(e, \Lambda)$ for any $e \in E$ even if it tries to eavesdrop on the communication between the sensors. After ϕ has been sent to MA and the protocol has finished, the data on the sensors should be erased. We describe the process of ensuring honest computation of ϕ in detail in Section 4.2.3 of the extended version [17] .

V. Discussion

The protocol requires minimal computational resources from the MA. Indeed, the computation of $g(\Lambda)$, and all data analysis therein, is conducted by the MPs. The MA only needs to construct an evaluation circuit C and zk-SNARK (S, V, P) for each of their queries g. In terms of data storage, the MA only needs to store the commitments σ to the demand and the total recorded volume of MP traffic ϕ for each data collecting period. If the Merkle Trees are built using the SHA256 hash function, then σ is only 256 bits, and is thus easy to store. ϕ is a single integer, which is also easy to store.

On the other hand, the hardware requirements for the Aggregated Roadside Audits may be difficult for cities to implement, as placing a sensor on every road in the city will be expensive. To address this concern, we present an alternative mechanism known as Randomized Roadside Audits (RRA) in Supplementary Material B.6 of the extended version [17] . RRA is able to use fewer sensors by randomly sampling the roads to be audited, however as a tradeoff for using fewer sensors, overreported demand will only be detected probabilistically. See Supplementary Material B.6 of the extended version [17] for more details.

There is a trade-off between privacy and diagnosis when using zero knowledge proofs. In the event that the zk-SNARK's verification function fails, i.e., $V(\sigma, z, \pi) = \texttt{False}$, we know that z is not a valid message, but we do not know *why* it is invalid. Specifically, $V(\sigma, z, \pi)$ does not specify which step of the evaluation algorithm C failed (See Stage 3 of Section IV-A). Thus in order to determine whether the failure was due to integrity checks, inconsistency between Λ and σ, or a mistake in the computation of g, further investigation would be required. Thus, while the zero knowledge proof enables us to check the correctness of z without directly inspecting the

data, it does not provide any diagnosis in the event that z is invalid.

VI. Conclusion

In this paper we presented an interactive protocol that enables a Municipal Authority to obtain insights from the data of Mobility Providers in a verifiable and privacy-preserving way. During the protocol, a Municipal Authority submits queries and a Mobility Provider computes responses based on its mobility data. The protocol is privacy-preserving in the sense that the Municipal Authority learns nothing about the dataset beyond the answer to its query. The protocol is verifiable in the sense that any deviation from the protocol's instructions by one party can be detected by the other. Verifiability is achieved by using cryptographic commitments and aggregated roadside measurements, and data privacy is achieved using zero knowledge proofs. We showed that the protocol can be generalized to a setting with multiple Mobility Providers using secure multi-party computation. We present a differentially private version of the protocol in Appendix I to address situations where the Municipal Authority has many queries.

There are several interesting and important directions for future work. First, while this work accounts for strategic behavior of the Municipal Authority and Mobility Providers, it assumes that drivers and customers will act honestly. A more general model which also accounts for potential strategic behavior of drivers and customers would be of great value and interest. Second, while secure multi-party computation can be used to generalize the protocol to settings with multiple Mobility Providers, generic tools for secure multi-party computation introduce computational and communication overhead. Developing specialized multi-party computation tools for mobility-related queries is thus of significant practical interest. Quantum computation is another promising method for developing faster cryptographic tools [28]. Finally, we suspect there are other applications for this protocol in transportation research beyond city planning and regulation enforcement that could be investigated.

References

[1] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*, vol. 3876, pp. 265–284, Springer, 2006.

[2] PCAST, "Big data and privacy: A technological perspective," 2014.

[3] M. Isaac, "How uber deceives the authorities worldwide," *New York Times*, Mar 2017.

[4] S. Szymkowski, "Google removes app that calculated if uber drivers were underpaid," *RoadShow*, Feb 2021.

[5] I. Dinur and K. Nissim, "Revealing information while preserving privacy," in *ACM Symposium on Principles of Database Systems*, pp. 202–210, ACM, 2003.

[6] R. W. van der Heijden, S. Dietzel, T. Leinmüller, and F. Kargl, "Survey on misbehavior detection in cooperative intelligent transportation systems," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 1, pp. 779–811, 2019.

[7] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, 2000.

[8] J. Hortelano, J. C. Ruiz, and P. Manzoni, "Evaluating the usefulness of watchdogs for intrusion detection in vanets," in *2010 IEEE International Conference on Communications Workshops*, pp. 1–5, 2010.

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

[10] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *2014 IEEE Symposium on Security and Privacy*, pp. 459–474, 2014.

[11] D. Gabay, K. Akkaya, and M. Cebe, "Privacy-preserving authentication scheme for connected electric vehicles using blockchain and zero knowledge proofs," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5760–5772, 2020.

[12] W. Li, C. Meese, H. Guo, and M. Nejad, "Blockchain-enabled identity verification for safe ridesharing leveraging zero-knowledge proof," *arXiv preprint arXiv:2010.14037*, 2020.

[13] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. D. Smith, "What can we learn privately?," *SIAM J. Comput.*, vol. 40, no. 3, pp. 793–826, 2011.

[14] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or A completeness theorem for protocols with honest majority," in *STOC 1987, New York, New York, USA* (A. V. Aho, ed.), pp. 218–229, ACM, 1987.

[15] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[16] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *FOCS 1985*, pp. 383–395, IEEE Computer Society, 1985.

[17] M. Tsao, K. Yang, S. Zoepf, and M. Pavone, "Trust but verify: Cryptographic data privacy for mobility management," *arXiv preprint*, 2021. Extended version. Available at `https://arxiv.org/abs/2104.07768`.

[18] Y. Sheffi, *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall, Englewood Cliffs, New Jersey, 1 ed., 1985.

[19] A. Gabizon, Z. J. Williamson, and O. Ciobotaru, "PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge," *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 953, 2019.

[20] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, "Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings," in *ACM Conference on Computer and Communications Security, CCS*, pp. 2111–2128, ACM, 2019.

[21] A. Chiesa, Y. Hu, M. Maller, P. Mishra, N. Vesely, and N. P. Ward, "Marlin: Preprocessing zksnarks with universal and updatable SRS," in *Advances in Cryptology - EUROCRYPT 2020*, vol. 12105 of *Lecture Notes in Computer Science*, pp. 738–768, Springer, 2020.

[22] B. Bünz, B. Fisch, and A. Szepieniec, "Transparent snarks from DARK compilers," in *Advances in Cryptology - EUROCRYPT 2020*, vol. 12105 of *Lecture Notes in Computer Science*, pp. 677–706, Springer, 2020.

[23] A. R. Block, J. Holmgren, A. Rosen, R. D. Rothblum, and P. Soni, "Time- and space-efficient arguments from groups of unknown order," in *Advances in Cryptology - CRYPTO 2021*, vol. 12828 of *Lecture Notes in Computer Science*, pp. 123–152, Springer, 2021.

[24] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more," in *IEEE Symposium on Security and Privacy, SP*, pp. 315–334, IEEE Computer Society, 2018.

[25] S. T. V. Setty, "Spartan: Efficient and general-purpose zksnarks without trusted setup," in *Advances in Cryptology - CRYPTO 2020*, vol. 12172 of *Lecture Notes in Computer Science*, pp. 704–737, Springer, 2020.

[26] M. Campanelli, D. Fiore, and A. Querol, "Legosnark: Modular design and composition of succinct zero-knowledge proofs," in *ACM Conference on Computer and Communications Security, CCS*, pp. 2075–2092, ACM, 2019.

[27] M. Campanelli, A. Faonio, D. Fiore, A. Querol, and H. Rodríguez, "Lunar: a toolbox for more efficient universal and updatable zksnarks and commit-and-prove extensions," *IACR Cryptol. ePrint Arch.*, p. 1069, 2020.

[28] L. Gyongyosi and S. Imre, "A survey on quantum computing technology," *Computer Science Review*, 2019.

[29] M. Blum, "Coin flipping by telephone - A protocol for solving impossible problems," in *COMPCON'82*, pp. 133–137, IEEE Computer Society, 1982.

[30] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology - CRYPTO* (J. Feigenbaum, ed.), vol. 576 of *Lecture Notes in Computer Science*, pp. 129–140, Springer, 1991.

APPENDIX I
INCORPORATING DIFFERENTIAL PRIVACY FOR THE LARGE QUERY REGIME

One potential concern with the protocol described in Section IV arises in the large query regime. It was shown in [5] that a dataset can be reconstructed from many accurate statistical measurements. One way to address this is to set a limit on the number of times the MA can query the data for a given time period. Such a restriction would not lead to data scarcity since the MP is collecting new data daily. Differential privacy offers a principled way to determine how many times MA should query a dataset (see Remark 12). Differentially private mechanisms address the result of [5] by reducing the accuracy of the responses to queries, i.e., responding to a query g with a noisy version of $g(\Lambda)$. In this section we describe how the protocol from section IV can be generalized to facilitate verifiable and differentially private responses from MP. To this end we first define differential privacy.

Definition 7 (Datasets and Adjacency)**.** A dataset Λ is a set of datapoints. In the context of transportation demand, a datapoint is the metadata corresponding to a single trip. We say two datasets Λ, Λ' are adjacent if either (a) $\Lambda \subset \Lambda'$ with Λ' containing exactly 1 more datapoint than Λ, or (b) $\Lambda' \subset \Lambda$ with Λ containing exactly 1 more datapoint than Λ'.

Definition 8 (Differential Privacy)**.** Let $\mathcal{F}$ be a σ-algebra on a space Ω. A mechanism $M : \mathcal{D} \to \Omega$ is (ϵ, δ)-differentially private if for any two adjacent datasets $\Lambda, \Lambda' \in \mathcal{D}$ and any $\mathcal{F}$-measurable event S,

$$\mathbb{P}\left(M(\Lambda) \in S\right) \leq e^{\epsilon}\mathbb{P}\left(M(\Lambda') \in S\right) + \delta.$$

In words, the output of a (ϵ, δ)-differentially private mechanism on Λ is statistically indistinguishable from the output of the mechanism on $\Lambda \cup \{\lambda\}$ for any single datapoint $\lambda \notin \Lambda$. Since Λ does not contain λ, $M(\Lambda)$ does not reveal any information about λ. Since $M(\Lambda \cup \{\lambda\})$ is statistically indistinguishable from $M(\Lambda)$, $M(\Lambda \cup \{\lambda\})$ does not reveal much about λ.

Example 9 (Laplace Mechanism for Vote Tallying)**.** Suppose a city is trying to decide whether to expand its railways or expand its roads based on a majority vote from its citizens. The dataset is $\Lambda := \{\lambda_1, ..., \lambda_n\}$ where λ_i is a boolean which is 0 if the ith citizen prefers the railway and 1 if the ith citizen prefers the roads. To implement majority vote, the city needs to compute $g(\Lambda) := \sum_{i=1}^{n} \lambda_i$. The Laplace Mechanism achieves $(\epsilon, 0)$-differential privacy for this computation via

$$M_{\text{laplace}}(\Lambda) := Y + \sum_{i=1}^{n} \lambda_i,$$

where Y has the discrete Laplace distribution: for any $k \in \mathbb{Z}$, $\mathbb{P}[Y = k] \propto e^{-\epsilon|k|}$. To see why this achieves $(\epsilon, 0)$-differential privacy, for any $1 \leq j \leq n$, note that

$$\frac{\mathbb{P}[M(\Lambda) = k]}{\mathbb{P}[M(\Lambda \setminus \{\lambda_j\}) = k]} = \frac{e^{-\epsilon\left|k - \sum_{i=1}^{n} \lambda_i\right|}}{e^{-\epsilon\left|k - \sum_{i \neq j} \lambda_i\right|}} \leq e^{\epsilon\lambda_j} \leq e^{\epsilon}.$$

Note that the noise distribution for Y depends only on ϵ, and is independent of n, the size of the dataset.

Remark 12 (Privacy Budget)**.** By composition rules, the result of k queries to a $(\epsilon, 0)$-differentially private mechanism is $(k\epsilon, 0)$-differentially private. Thus a dataset should only be used to answer k separate $(\epsilon, 0)$-differentially private queries if $e^{k\epsilon}$ is sufficiently close to 1.

A. Goal: Differential Privacy without Trust

Given a query function g from MA, let M be an polynomial-time computable (ϵ, δ)-differentially private mechanism for computing g. For a given dataset Λ we can represent the random variable $M(\Lambda)$ with a function $\widetilde{g}(\Lambda, Z)$ where $Z \in \{0,1\}^v$ represents the random bits used by M. Here v is an upper bound on the number of random bits needed for the computation of M. By its construction, $\widetilde{g}(\Lambda, Z)$ is (ϵ, δ)-differentially private if Z is drawn uniformly at random over $\{0,1\}^v$. Therefore differential privacy is achieved if MP draws Z uniformly at random over $\{0,1\}^v$ and sends $\widetilde{g}(\Lambda, Z)$ to MA. However, as mentioned in Assumption 1, we are studying a model where MP can act strategically. Thus we cannot assume that MP will sample Z uniformly at random if there is some other distribution over Z that leads to a more favorable outcome for MP. We revisit Example 9 to illustrate this concern.

Example 10 (Dishonest Vote Tallying)**.** Consider the setting from Example 9. The Laplace mechanism can be represented as

$$\widetilde{g}(\Lambda, Z) := Y + \sum_{i=1}^{n} \lambda_i, \text{ where } Y = F^{-1}_{\text{laplace}}\left(\frac{\text{int}(Z)}{2^v}\right),$$

where $\text{int}(Z)$ is the integer whose binary representation is the bits of Z. Here F^{-1}_{laplace} is the inverse cumulative distribution for the discrete Laplace distribution. Thus $F^{-1}_{\text{laplace}}(\text{int}(Z)/2^v)$ is an application of inverse transform sampling that converts a uniform random variable Z into a random variable Y with a discrete Laplace distribution. Suppose the MP has a ridehailing service and would thus prefer an upgrade to city roads over an upgrade to the railway system. If this is the case, choosing Z so that $\widetilde{g}(\Lambda, Z) > n/2$ (as opposed to choosing Z randomly) is a weakly dominant strategy for MP, even if $g(\Lambda) < n/2$ and a majority of the citizens prefer railway upgrades.

Thus we need a way to verify that the randomness Z used in MP's evaluation of $g(\Lambda, Z)$ has the correct distribution. We will now show how the protocol can be adjusted to accommodate this, and as a consequence, enable verifiable differentially private data queries for MA.

Remark 13 (MA provided randomness)**.** One natural attempt to ensure that Z is uniformly random is to have MA specify Z. However, this destroys the differential privacy, since for some mechanisms (including the Laplace mechanism) $g(\Lambda)$ can be computed from $\widetilde{g}(\Lambda, Z)$ and Z. Also, it is not clear a priori whether such a setup is strategyproof for MA.

B. A Differentially Private version of the protocol

In this section, we present modifications to the protocol from Section IV-A that enables verifiable differentially private

responses from MP. At a high level, the MA and MP jointly determine the random bits Z via a coin flipping protocol [29]. The zk-SNARK can then be modified to ensure that $\widetilde{g}(\Lambda, Z)$ is computed correctly. The protocol has a total of 6 stages which are described below.

Stage 0: (Data Collection) MP builds a Merkle Tree T_Λ of the demand Λ that it serves. It computes a commitment $\sigma := \mathsf{MCommit}(\Lambda, r)$ to this demand. Additionally, MP samples Z_{mp} uniformly at random from $\{0,1\}^v$ and computes a Pedersen commitment [30] $z_{\text{mp}} := \mathsf{Commit}(Z_{\text{mp}}, r_{\text{mp}})$. The Pederson commitment scheme is a secure commitment scheme which is perfectly hiding and computationally binding. MP sends both σ, z_{mp} to MA.

Stage 1: (Integrity Checks) Same as in Section IV-A.

Stage 2: (Message Specifications) MA specifies the function g it wants to compute. Additionally, MA samples Z_{ma} uniformly at random from $\{0,1\}^v$ and specifies a differentially private mechanism $\widetilde{g}$ for the computation of g.

Stage 3: (zk-SNARK Construction) MA constructs an evaluation circuit C for the function $\widetilde{g}$. The public parameters of C are $\sigma, z_{\text{mp}}, Z_{\text{ma}}, z$ and the input to C is a witness of the form $w = (\Lambda_w, r_w, c_w, Z_{\text{mp},w}, r_{\text{mp},w})$. C does the following:

1) Checks whether the Rider Witness and Aggregated Roadside Audit tests are satisfied,
2) Checks whether $\mathsf{MCommit}(\Lambda_w, r_w) = \sigma$,
3) Checks whether $\mathsf{Commit}(Z_{\text{mp},w}, r_{\text{mp},w}) = z_{\text{mp}}$,
4) Checks whether $\widetilde{g}(\Lambda_w, Z_{\text{ma}} \oplus Z_{\text{mp},w}) = z$. (Here $\oplus$ is bit-wise XOR.)

C will return `True` if and only if all of these checks pass. MA constructs a zk-SNARK (S, V, P) for C and sends $g, \widetilde{g}, Z_{\text{ma}}, C, (S, V, P)$ to MP.

Stage 4: (Function Evaluation) If $\widetilde{g}$ is a differentially private mechanism for computing g, then MP computes a message $z = \widetilde{g}(\Lambda, Z_{\text{ma}} \oplus Z_{\text{mp}})$ and a witness $w := (\Lambda, r, c_w, Z_{\text{mp}}, r_{\text{mp}})$ to the correctness of z.

Stage 5: (Creating a Zero Knowledge Proof) Same as in Section IV-A.

Stage 6: (zk-SNARK Verification) Same as in Section IV-A.

In Supplementary Material B.7 of the extended version [17] we show that this protocol has the following two desirable features that enable verifiable and differentially private responses from MP to MA queries.

1) Verifiability - If the MA receives a valid proof from MP, then it can be sure that the corresponding message is indeed $\widetilde{g}(\Lambda, Z_{\text{ma}} \oplus Z_{\text{mp}})$.
2) Differential Privacy - The MP's output is differentially private with respect to the dataset Λ if at least one of $Z_{\text{ma}}, Z_{\text{mp}}$ is sampled uniformly at random.

Remark 14 (A note on Local Differential Privacy)**.** Local Differential Privacy [13] addresses the setting where the data collector is untrusted. Differential privacy is achieved by users adding noise to their data before sending it to the data collector. This is in contrast to the setting we study here where an untrusted data collector has the clean data of many users. We chose to study the latter model due to the way current mobility companies collect high resolution data on the trips they serve. Additionally, local differential privacy requires users to add noise to their data so they become statistically indistinguishable from one another. In the context of transportation, this means the noisy data of users will be statistically indistinguishable from one another, even if they have very different travel preferences. This level of noise significantly reduces the accuracy of any computation done on the data.

Matthew Tsao is a Ph.D student in the department of Electrical Engineering at Stanford University. Prior to Stanford, he obtained a BS in Electrical Engineering with highest honors from the University of Illinois at Urbana Champaign. Matt's current research involves developing robust algorithms for control of fleets of autonomous cars using ideas from optimization, online algorithms, and statistics.

Kaidi Yang is a Postdoctoral Scholar with the Autonomous Systems Lab in the Department of Aeronautics and Astronautics at Stanford University. He obtained the Ph.D. degree in Civil Engineering from ETH Zurich in 2019. Prior to that, he received a BEng. in Automation, a BSc. in Pure and Applied Mathematics (minor), and an MSc. in Control Science and Engineering from Tsinghua University in China. His research interest lies in multimodal traffic operation with emerging technologies and shared mobility.

Stephen Zoepf is the Chief of Policy Development for Ellis & Associates, where he helps guide the development of open-source software products for cities to manage modern transportation systems. He holds a Ph.D., M.Sc. and B.Sc. from MIT and has two decades of experience in transportation and mobility. Stephen previously led the Center for Automotive Research at Stanford as Executive Director. His research has been covered in numerous popular press articles, initiated a Congressional probe, and has been lampooned in The Onion.

Marco Pavone is an Associate Professor of Aeronautics and Astronautics at Stanford University and the Director of Autonomous Vehicle Research at NVIDIA. He received a Ph.D. degree in Aeronautics and Astronautics from the Massachusetts Institute of Technology in 2010. His main research interests are in the development of methodologies for the analysis, design, and control of autonomous systems, with an emphasis on self-driving cars, autonomous aerospace vehicles, and future mobility systems.