Hindawi
Security and Communication Networks
Volume 2021, Article ID 6619770, 16 pages
https://doi.org/10.1155/2021/6619770

Research Article

Privacy-Preserving Multipoint Traffic Flow Estimation for Road Networks

Elmahdi Bentafat, M. Mazhar Rathore, and Spiridon Bakiras

Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar

Correspondence should be addressed to Spiridon Bakiras; sbakiras@hbku.edu.qa

Received 29 November 2020; Revised 11 February 2021; Accepted 18 March 2021; Published 1 April 2021

Academic Editor: Flavio Lombardi

Intelligent transportation systems necessitate a fine-grained and accurate estimation of vehicular traffic flows across critical paths of the underlying road network. However, such statistics should be collected in a manner that does not disclose the trajectories of individual users. To this end, we introduce a privacy-preserving protocol that leverages roadside units (RSUs) to communicate with the passing vehicles, in order to construct encrypted Bloom filters stemming from random vehicle IDs that are chosen secretly by the individual vehicles. Each Bloom filter represents the set of vehicle IDs that contacted the RSU but may also be used to estimate the traffic flow between any number of RSUs. More precisely, we designed a probabilistic model that approximates multipoint traffic flows by estimating the number of common vehicles among a given set of RSUs. Through extensive simulation experiments, we demonstrate that our protocol is very accurate—with a minor deviation from the real traffic flow—and show that it reduces the estimation error by a large factor, when compared to the current state-of-the-art approaches. Furthermore, our implementation of the underlying cryptographic primitives illustrates the feasibility, practicality, and scalability of the system.

1. Introduction

Traffic statistics facilitate transportation authorities in many use cases, including investment plans, signal time determination, road expansions, etc. Traffic statistics are measured in terms of single-point, point-to-point, or multipoint traffic flows. Single-point traffic flow refers to the number of vehicles passing through a specific location, which can be measured by placing fixed sensors at roadsides, such as inductive loop detectors, wireless magnetometer sensors, road cameras, or microwave radar sensors. Single-point statistics are useful for measuring the average annual daily traffic (AADT) [1–4]. On the other hand, point-to-point traffic flow, sometimes referred to as origin-destination (O-D) flow, is defined as the total number of vehicles moving from one location to another. Finally, multipoint traffic refers to the traffic flow that passes through a set of specific locations. Point-to-point and multipoint traffic statistics may be deduced from single-point counters [5]; however, there are serious concerns about the accuracy of such approaches because they are oblivious to the identities of the underlying vehicles.

Consequently, to derive accurate traffic statistics, we need alternative methods for collecting data from moving vehicles. One approach is to use the drivers' smartphones [6, 7] or the GPS systems integrated in most vehicles [8, 9], in order to generate detailed object trajectories. Alternatively, recent advancements in vehicular communications and networking technologies have brought cyber physical systems (CPS) into road networks, where roadside units are used to collect traffic data [10]. The dedicated short-range communications (DSRC) protocol is standardized by the IEEE (802.11p) [11] and enables the direct communication between vehicles and roadside units (RSUs). In this scenario, generating accurate traffic statistics is trivial: each vehicle reports its ID to every RSU it encounters, with all the reports being aggregated to a centralized server (the transportation authority). Nevertheless, this approach violates the privacy of the vehicle owners and may reveal sensitive personal information, such as home and work locations, habits, etc.

To address these privacy concerns, Zhou et al. [12, 13] have proposed the use of a bit array as an alternative to individual vehicle IDs. In particular, every vehicle selects (in advance) a set of s bit locations that it may reveal to an RSU. When the vehicle identifies a new RSU, it randomly selects one of the s locations and sends it to the RSU as its identifier. Each RSU gradually aggregates the bit array data from all passing vehicles by setting a bit to "1" if it is selected by at least one vehicle (nonselected bits are set to "0"). Point-to-point [13] or multipoint [12] traffic flows are then constructed by comparing the bit arrays of the corresponding RSUs. While this approach improves the privacy compared to the trivial method, it still has several limitations. First, the bit information is exchanged in plaintext and is, thus, vulnerable to timing attacks. Indeed, any number of RSUs may collude to deduce the vehicles' secret information (the s bit locations), by correlating successive samples based on the driving distance between the RSUs. Second, for sufficient privacy, s should be relatively large (e.g., $s \geq 5$), which negatively affects the accuracy of the traffic flow statistics.

To this end, our work advances the state of the art in two directions. Our major contribution is a novel method that derives from our earlier conference paper [14] and summarizes the vehicle information at each RSU into an encrypted Bloom filter [15]. We employ a two-tiered approach where (i) the vehicles' Bloom filters are encrypted with a simple onetime pad (OTP) cipher and (ii) the OTP keys are encrypted with a homomorphic *threshold* public key cryptosystem. The underlying cryptosystems make it infeasible for an adversary to decrypt the individual Bloom filters, thus enhancing significantly the vehicles' privacy. Each RSU aggregates the Bloom filters and OTP keys from multiple vehicles and sends the corresponding ciphertexts to the transportation authority. Finally, the transportation authority engages multiple trusted parties to decrypt the aggregate OTP keys and retrieve the plaintext Bloom filters.

The second contribution is the extension of our previous work [14] from origin-destination to multipoint traffic flow estimation. This is an important feature, because it allows transportation authorities to estimate the traffic flows across arbitrary road network paths. Specifically, we introduce a simple and accurate probabilistic method for estimating the number of common vehicles among any number of distinct Bloom filters, which is an indication of the traffic flow volume between the underlying RSUs. Furthermore, we significantly extended our simulation experiments, by considering more diverse settings where the number of vehicles at each RSU exhibits a large variance. Our results demonstrate that, compared to the current state-of-the-art approaches, our methods improve the accuracy of both point-to-point and multipoint traffic flow estimation by a large factor. In addition, our software implementation of the basic cryptographic primitives at the RSUs and the server illustrates the feasibility and scalability of our approach.

The remainder of this paper is organized as follows. Section 2 surveys the existing literature on traffic flow estimation. Section 3 discusses the data structures and cryptographic primitives utilized in our work, and it also presents the underlying system and threat models. Section 4 introduces our privacy-preserving aggregation protocol, and Section 5 presents the probabilistic model for estimating the traffic flows. Section 6 shows the results of our experimental evaluation, and Section 7 concludes our work.

2. Related Work

Estimating and predicting traffic flow is a mature research problem that has evolved over time. Early work on road network traffic flows focused on the prediction of the annual average daily traffic (AADT). To this end, a variety of machine-learning models have been applied. Mohammed et al. [4] predicted the AADT on county roads in the state of Indiana, USA, using a linear regression model. Lam and Xu [3] estimated the AADT on urban roads of Hong Kong with a neural network, using short counts. A similar prediction model was designed by Eom et al. [2], who computed the spatial dependencies with the use of a regression method. Specifically, the authors leveraged a geostatistical technique, called *kriging*, for modeling spatial trends and spatial correlations between monitoring stations and neighboring stations. Neto et al. [1] utilized support vector machine for regression (SVR), by applying data-dependent parameters based on the distribution of the training data. Similarly, Tsapakis et al. [16] designed twelve models based on regression and Bayesian analysis, using various parameters (such as roadway functional class, population density, and spatial location) collected from five regions in the state of Ohio, USA. Finally, other research work has addressed the feature selection problem for AADT prediction. For example, Yang et al. [17] selected the AADT estimation parameters via the smoothly clipped absolute deviation (SCAD) penalty. All the above systems exploit the capabilities of either sensors or roadside units for traffic data collection.

On the other hand, the authors of [6, 7, 9] utilize data from mobile phones and GPS devices to extract origin-destination information. They exploited the global system for mobile communications (GSM) to observe the flow of mobile phones using cellular network technology, and correlate it to the road network traffic flow. This is similar to the approach used by Google Maps and Waze (https://www.google.com/maps, https://www.waze.com/) to optimize their routing decisions [8].

Hoh et al. [18] highlighted the serious privacy threats in traffic monitoring systems, as they were able to locate 85% of the drivers' home locations from the collected data. This is also true for popular apps, like Google Maps and Waze, which use a static ID for each reporting client, even across different trips [8]. Therefore, to protect the privacy of trajectory data, Hoh and Gruteser [19] proposed a path perturbation algorithm for a centralized, trusted server, by employing path confusion. PADAVAN [20] is another scheme for anonymous data collection that allows anonymous data reporting, while avoiding fake submissions and linkage between submitted samples. Likewise, Rass et al. [21] introduced trajectory anonymization by deriving pseudonyms for trips and samples. Finally, Hoh et al. [22] proposed a distributed and privacy-preserving traffic monitoring

system that utilizes virtual trip lines, i.e., geographical markers where the vehicle has to report its location. Based on the trip line ID, the scheme allows for aggregation and cloaking of several locations, without revealing the precise locations. With a distributed architecture, no single entity has full information of probe IDs and fine-grained locations. However, if multiple compromised entities collaborate, location updates may be revealed with high accuracy.

The abovementioned schemes introduce some level of privacy in traffic monitoring systems, but they are not well-suited toward estimating point-to-point or multipoint traffic flows with high accuracy. To this end, several researchers applied intricate cryptographic protocols to protect the privacy of drivers and their trips. Forster et al. [23] proposed a distributed secret sharing algorithm, using location- and time-specific keys, and enforced k-anonymity of location data in a decentralized environment. At the end of a trip, the vehicle reports its O-D locations with the corresponding time information. Trip reports are encrypted with location and time-specific keys, and uploaded to a centralized and untrusted database. The distributed key sharing algorithm assures that trip reports are available once k vehicles have the same trip with matching O-D and start-end times. Zhou et al. [24] measured origin-destination traffic flows using commutative one-way hash functions that are constructed from an RSA-like cryptosystem. Although the hash function hides the vehicle's ID from the RSU, it fails to protect the privacy of the trajectory when all the data are aggregated at the centralized server.

The current state-of-the-art protocols for privacy-preserving traffic data collection are due to Zhou et al. [12, 13] and Sun et al. [25, 26], which provide privacy without the use of cryptographic techniques. Specifically, in their methods, every receiver maintains a physical bit array of size m (similar to a Bloom filter), and each passing vehicle sets one of the bits to "1". To avoid being identified across multiple receivers, the vehicle uses a logical bit array of size $s \ll m$, i.e., it can only set (randomly) one of s preselected bits at each receiver. Zhou et al. [12] applied maximum-likelihood estimation (MLE) on the bit arrays to measure the intensity of the traffic flow between any pair of receivers. In their other work [13], the authors introduced variable sized bit arrays to estimate the traffic flow across multiple RSUs. Using the same data collection approach, Sun et al. [25] computed the number of vehicles that persistently (in every measurement period) travel on a given road network edge. Furthermore, the same authors [26] also analyzed the number of vehicles that travel x out of T measurement periods on the network edge.

While the aforementioned data collection approach is very efficient, it has several shortcomings. For sufficient privacy, s should be large, but this negatively affects the accuracy of the traffic flow estimation. In addition, the lack of encryption makes it possible to launch a variety of attacks, such as timing or direct observation, which may disclose the contents of a vehicle's logical bit array. Our method overcomes these limitations by (i) revealing only aggregate Bloom filter data and (ii) utilizing encryption for the submission of individual measurements.

3. Preliminaries

3.1. Bloom Filter. A Bloom filter is a fast, memory-efficient, and probabilistic data structure that was designed by Burton Howard Bloom [15] in 1970 for the purpose of rapid searching (set membership). It is essentially a bit array of size m, whose bits are initially set to "0." Prior to adding elements into the Bloom filter, we must define k hash functions $H_1, H_2, \ldots, H_k$, each returning a value in the range $[0, m)$. Then, to add an element v into the Bloom filter, we derive k random indices by hashing v with each of the k hash functions, i.e., $i_1 = H_1(v)$, $i_2 = H_2(v)$, ..., $i_k = H_k(v)$. Finally, for each computed index i_j, we set the corresponding bit on the bit array to "1." Similarly, to search for an element v in the Bloom filter, we first derive the k indices $i_1, i_2, \ldots, i_k$ from the hash functions and, if at least one of the bits in these locations is "0," v is definitely not a member of this set. Otherwise, v is most likely included in the set, although there is a small probability for a false positive because of collisions (multiple elements may share the same index).

To reduce the false-positive rate, the optimal value for m must be selected based on the number k of hash functions, and the expected number of elements that will be inserted into the Bloom filter. Note that a larger k increases the complexity of the Bloom filter but, in our protocol, a large k also results in more user privacy. It is also worth noting that it is possible to produce the Bloom filter of the union of two (or more) sets, by computing the bitwise OR of the individual Bloom filters. We will take advantage of this property in our methods described in Section 5.

3.2. Paillier Cryptosystem. Public-key homomorphic cryptosystems [27] allow for the manipulation (through mathematical operations) of encrypted plaintexts without requiring access to the private decryption key. In particular, fully homomorphic encryption (FHE) [28] permits arbitrary computations over ciphertexts (both additions and multiplications) and can, thus, be used with any circuit over encrypted data. However, FHE is still very inefficient and not suitable for real-time traffic data collection. Instead, in our work, we rely on Paillier's additively homomorphic cryptosystem [29]. Paillier encryption is semantically secure and its security is based on the decisional composite residuosity assumption (DCRA). The scheme works as follows.

3.2.1. Key Generation. Select two large primes p, q of equal length and compute the RSA modulus $n = pq$. Choose a uniformly random integer $g \in \mathbb{Z}^*_{n^2}$ and compute $\lambda = \text{lcm}(p-1, q-1)$ and $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$, where $L(x) = (x-1)/n$. Output public key (n, g) and the corresponding private key (λ, μ).

3.2.2. Encryption. Given a message $m < n$, choose a uniformly random integer $g \in \mathbb{Z}^*_n$ and output ciphertext $c = g^m r^n \bmod n^2$.

3.2.3. Decryption. Given a ciphertext c, compute the plaintext message $m = L\,(c^{\lambda} \bmod n^2) \cdot \mu \bmod n$.

To illustrate the additively homomorphic property of the Paillier cryptosystem, consider the encryption of two messages $\mathrm{Enc}(m_1)$ and $\mathrm{Enc}(m_2)$. Then, the following equation holds:

$$\begin{aligned}\mathrm{Enc}(m_1) \cdot \mathrm{Enc}(m_2) &= (g^{m_1} r_1^n)(g^{m_2} r_2^n) \bmod n^2, \\ &= g^{m_1+m_2} (r_1 r_2)^n \bmod n^2, \\ &= \mathrm{Enc}(m_1 + m_2).\end{aligned} \tag{1}$$

In our work, we leverage a threshold variant of Paillier's encryption scheme [30], where the secret key is shared among t parties. While this is easily done with the help of a trusted third party, there are protocols that allow the t parties to compute the Paillier key pair in a distributed manner. With the threshold cryptosystem, all t parties must cooperate to decrypt a Paillier ciphertext.

3.3. System Model. We consider the cyber physical system depicted in Figure 1, where roadside units (also called receivers) are installed at different geographical locations along the road network. Every vehicle and receiver is equipped with a computing unit, and is able to communicate with other devices through the IEEE 802.11p protocol. Whenever a vehicle comes in close proximity to an RSU (e.g., within 100–500 m), it transmits an encrypted Bloom filter that is a representation of its unique identifier (to enhance privacy, a vehicle may choose a different identifier at the start of a new trip.). The RSU blindly aggregates the individual Bloom filters and submits the resulting ciphertexts to the transportation authority. Note that we are interested in collecting *fine-grained* traffic statistics, so each RSU will produce a new Bloom filter when (i) a timer expires (e.g., every 5–10 minutes) or (ii) a threshold number of measurements is reached (e.g., 1000–2000 vehicles). To satisfy basic privacy requirements, an RSU will not submit a Bloom filter if the number of vehicles is below a lower threshold (e.g., 100–200 vehicles). Such fine-grained measurements allow for the collection of important traffic statistics, including point-to-point or multipoint travel speed estimation.

Prior to system deployment, the transportation authority initializes a threshold Paillier cryptosystem in collaboration with several third-party trusted entities. For example, these entities may include various consumer protection agencies and other nonprofit organizations. The reason for employing a threshold cryptosystem is to prevent individual parties—especially the transportation authority that has access to all data through the RSUs—from decrypting the Bloom filters that are transmitted by the passing vehicles. Threshold decryption necessitates the collaboration of all involved parties, so a single honest player is sufficient for ensuring that only aggregate Bloom filters are being decrypted.

3.4. Threat Model. We assume that all entities in the aforementioned system model are semi-honest, i.e., they will execute all protocols correctly but will try to gain an advantage (in identifying vehicle-specific secret information) by examining the exchanged messages. We also allow collusions among the transportation authority and the RSUs, i.e., the adversary is given the full communication transcript of the underlying network. Furthermore, our methods do not require TLS-based communications to thwart eavesdropping attacks, because all transmitted information is encrypted (nevertheless, if we wish to authenticate the vehicles and/or receivers, we may utilize the TLS protocol with the existing public key infrastructure.) The only requirement with regard to privacy is that, out of the t trusted parties engaged in the threshold cryptosystem, at most $t-1$ of those can collude. Finally, we assume that the vehicles can remove all identifying information when communicating with a receiver, for e.g., by spoofing the actual MAC address of their network interface card. While it is still possible to track a vehicle using road cameras or other devices, such attacks are out of the scope of this paper.

4. Privacy-Preserving Data Aggregation

Our solution comprises three distinct phases: Bloom filter encryption, aggregation, and decryption. We describe all of them in detail in the following sections.

4.1. Bloom Filter Encryption. To add a vehicle v into the Bloom filter, we compute $H_i(v)$, $\forall i \in \{1, 2, \ldots, k\}$ and set the corresponding bits in the bit vector to "1." Unfortunately, aggregating Bloom filters with an additively homomorphic public key encryption scheme (such as Paillier) is infeasible, because the logical OR operation necessitates a fully homomorphic cryptosystem [28]. Instead, we may utilize *counting* Bloom filters [31] which are integer vectors that enable element deletions as well. In a counting Bloom filter, instead of setting the bits at the k vector positions, we simply increment the corresponding counters. However, this approach is vulnerable to correlation attacks, by inspecting the Bloom filters of two or more successive receivers. If their counters differ in only a small fraction of the m vector locations, an adversary may be able to deduce the k secret indexes of the noncommon vehicles.

As a result, in this work, we employ a onetime pad cipher to encrypt the Bloom filter vectors. In particular, let $q = 2^w$ be a prime power and let Fq be the finite field of integers modulo q. For a vehicle v, its Bloom filter is represented as a vector $\mathbf{b} \in F_q^m$, where b_i, $\forall i \in \{H_1(v), H_2(v), \ldots, H_k(v)\}$ is chosen uniformly at random from the range $[1, q)$. The remaining values are all set to zero. To encrypt its Bloom filter, the vehicle chooses a random vector $\mathbf{e} \leftarrow F_q^m$ and outputs the following ciphertext (in modulo q arithmetic).

$$\mathbf{c} = \mathbf{b} + \mathbf{e}. \tag{2}$$

The next step is to devise an efficient method for vehicles to communicate the *aggregate* encryption keys to the transportation authority. For that purpose, we employ the additively homomorphic Paillier cryptosystem

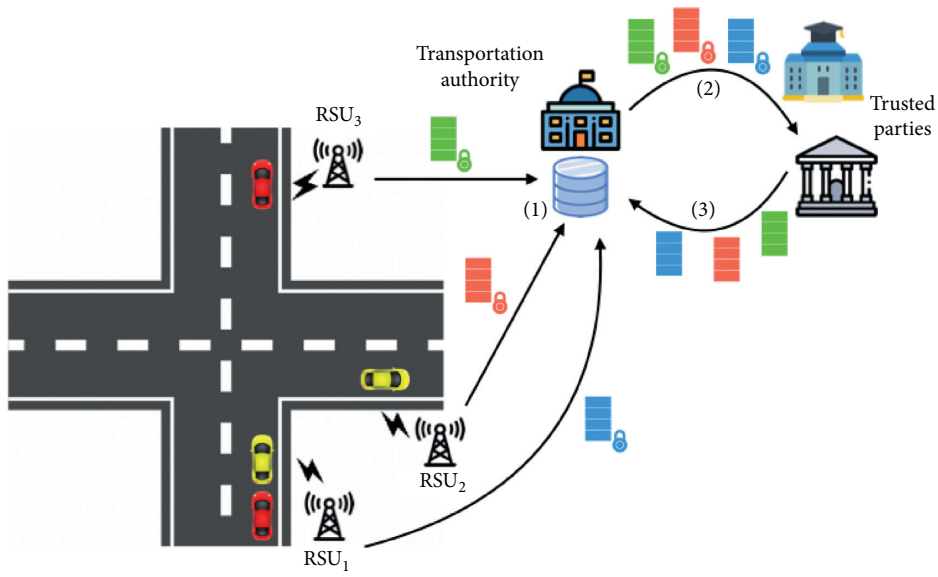

Figure 1: System model.

that is instantiated prior to system deployment. Assume now that the number of vehicles that may be summarized into a single Bloom filter is upper bounded by n. Then, the number of bits required to store a single Bloom filter entry is $\log n + \log q$. Based on the maximum message size that can be encrypted under Paillier (which depends on its RSA composite), we denote as l the max number of Bloom filter entries that can fit into a single Paillier ciphertext. Then, the vehicle will output the following ciphertext vector $\mathbf{r}$, where "|" denotes the concatenation operator.

$$\mathbf{r} = \left[\mathrm{Enc}\left(e_0|e_1|...|e_{l-1}\right), ..., \mathrm{Enc}\left(...|e_{m-2}|e_{m-1}\right)\right]^T. \quad (3)$$

The size of vector $\mathbf{r}$ is m/l and the elements e_i represent the elements of the key vector $\mathbf{e}$. The tuple $\langle \mathbf{c}, \mathbf{r} \rangle$ is the encrypted Bloom filter (i.e., identification) of that vehicle.

4.2. Bloom Filter Aggregation. When the vehicle encounters a new RSU, it will transmit its Bloom filter $\langle \mathbf{c}, \mathbf{r} \rangle$. After that, it will compute a re-randomized version of the Bloom filter, by choosing fresh random values for vectors $\mathbf{b}$ and $\mathbf{e}$. In this way, the vehicle cannot be identified across multiple RSUs. Each RSU maintains, locally, an aggregate (encrypted) Bloom filter $\langle \mathbf{c}^A, \mathbf{r}^A \rangle$ that summarizes the vehicles that have passed during the current measurement period. Once it receives a new sample from a passing vehicle, it updates the vectors as follows:

$$\begin{aligned} \mathbf{c}^A &= \mathbf{c}^A + \mathbf{c}, \\ \mathbf{r}^A &= \mathbf{r}^A \odot \mathbf{r}, \end{aligned} \quad (4)$$

, where $\odot$ denotes element-wise multiplication. When the current measurement period ends, the RSU will send $\langle \mathbf{c}^A, \mathbf{r}^A \rangle$ to the transportation authority, along with the duration of the measurement period (start and end time).

4.3. Bloom Filter Decryption. Decryption is a two-step process that involves the transportation authority and all the trusted third-parties. Once the transportation authority receives a new encrypted Bloom filter $\langle \mathbf{c}^A, \mathbf{r}^A \rangle$, it engages all t trusted parties to collectively decrypt the aggregate encryption key $\mathbf{e}^A = \sum_{i=1}^{n} \mathbf{e}^i$ from the ciphertext vector $\mathbf{r}^A$. This step entails the threshold decryption of m/l Paillier ciphertexts. Next, it reduces (element-wise) $\mathbf{e}^A$ modulo q, and computes the plaintext of the aggregate Bloom filter as follows:

$$\mathbf{b}^A = \mathbf{c}^A - \mathbf{e}^A. \quad (5)$$

It is important to note that, an adversary cannot determine the number of vehicles that have set a certain bit, because the corresponding value is uniformly random in the range $[0, q)$. The downside of this approach is that certain Bloom filter entries that have been selected by *at least* two vehicles may produce an incorrect value of zero (which signifies a "0" bit). However, the probability of that event is low. More specifically, let $P(i)$ be the probability that a certain Bloom filter entry is selected by exactly i out of n vehicles:

$$P(i) = \binom{nk}{i} \left(\frac{1}{m}\right)^i \left(1 - \frac{1}{m}\right)^{nk-i}, \quad (6)$$

where m is the Bloom filter size, and k is the number of hash functions. From this formula, we may compute the bit error probability P_{err} as follows:

$$P_{err} = [1 - P(0) - P(1)] \frac{1}{q}. \quad (7)$$

As an example, if $n = 2000$, $m = 8000$, $k = 4$, and $q = 2^{10}$, the bit error probability is just 0.026%. As we will show in our simulation experiments, the effect of bit errors on the accuracy of our protocol is negligible.

4.4. Privacy Analysis. We define as a privacy breach the disclosure of a vehicle's secret Bloom filter. This may be accomplished by (i) performing ciphertext-only attacks on the underlying cryptosystems or (ii) analyzing Bloom filters from different receivers.

Regarding the first type of attack, we argue that it is infeasible because of the semantic security of the OTP and Paillier cryptosystems that render every message indistinguishable. Furthermore, according to our threat model stated in Section 3.4, at least one of the trusted third-parties will not collude to decrypt individual Bloom filters. Instead, the only plaintext information available to the adversary is the aggregate encryption key $\mathbf{e}^A = \sum_{i=1}^{n} \mathbf{e}^i$ from the OTP ciphertexts of the n vehicles. That information alone is not sufficient to decrypt individual Bloom filters.

Indeed, let us consider a single Bloom filter entry j, and the n ciphertext values that are known by the adversary, namely c_1, c_2, ..., c_n. To retrieve the corresponding plaintext values b_i, the adversary must solve the following system of equations:

$$\begin{aligned} c_i &= b_i + e_i - s_i \cdot q,\ \forall i \in \{1, 2, \ldots, n\}, \\ e_1 + e_2 + \cdots + e_n &= e_j^A. \end{aligned} \tag{8}$$

Clearly, a unique solution does not exist, since there are $n+1$ equations with $3n$ unknowns. In fact, we can easily produce a solution for any combination of vehicles that have selected a nonzero value for that exact Bloom filter entry.

Therefore, the only viable attack vector for an adversary is to examine the Bloom filters from two different RSUs, whose Hamming distance is small. To this end, we consider the worst-case scenario for our approach, which entails two almost identical datasets. In particular, consider the unlikely scenario where the adversary has knowledge (e.g., using external observations) that sets A and B (containing $n-1$ and n vehicles, respectively) are constructed such that all of A's vehicles are also present in B (this is similar to the definition of differential privacy [32]). We can then determine the probability that the adversary can derive one or more of the k bit locations that identify the extra vehicle. Let P_i be the probability that we can identify exactly i out of k bits. This is equal to the probability that the i bits have been set by exactly one vehicle, which, using equation (7), can be written as

$$P_i = P(1)^i = \left[\left(1 - \frac{1}{m}\right)^{nk}\left(\frac{nk}{m-1}\right)\right]^i. \tag{9}$$

For instance, if $n = 2000$, $m = 8000$, and $k = 4$, the probability of recovering the vehicle's entire Bloom filter is just 1.8%.

5. Traffic Flow Estimation

Let A_1, A_2, ..., A_N be N sets containing the vehicles that contacted receivers RSU_{A1}, RSU_{A2}, ..., RSU_{AN}, respectively. Also, let $|A_i|$ denote the cardinality of A_i. Using the basic set theory, the number of common vehicles across the N RSUs (i.e., the traffic flow on that specific road network path) can be computed as follows:

$$\left|\cap_{i=1}^{N} A_i\right| = \sum_{i=1}^{N}|A_i| + (-1)^3 \sum_{1 \le i_1 < i_2 \le N} \left|A_{i_1}\ U\ A_{i_2}\right| + \cdots\cdots + (-1)^{N+1} \sum_{1 \le i_1 < i_2 < \ldots < i_N \le N} \left|A_{i_1}\ U\ A_{i_2}\ U\ \ldots\ A_{i_N}\right|. \tag{10}$$

For example, if we consider $N = 3$ RSUs, then the number of common vehicles is equal to:

$$\begin{aligned} |A_1 \cap A_2 \cap A_3| = {} & |A_1| + |A_2| + |A_3| - |A_1 \cup A_2| \\ & - |A_1 \cup A_3| - |A_2 \cup A_3| + |A_1 \cup A_2 \cup A_3|. \end{aligned} \tag{11}$$

Nevertheless, in our traffic monitoring scenario, no information is available regarding the vehicle IDs that are present in each of the above sets, and therefore, it is not possible to compute an exact solution. Specifically, even though the individual cardinalities $|A_i|$ can be disclosed by the corresponding RSUs, this is not advisable in terms of privacy, because it contributes to the adversary's knowledge. On the other hand, it is infeasible to compute the exact cardinality for any of the set unions. As such, we can only rely on cardinality estimations that may be derived from the individual Bloom filters.

Recall, from Section 3.1, that we can compute the correct Bloom filter of the union of an arbitrary number of distinct Bloom filters, by applying the logical OR operation on the corresponding bit arrays. Therefore, to estimate the traffic flow across any number of receivers, we need a formula that estimates the number of elements stored in a Bloom filter, based on the number of bits that are set. To this end, we employ equation (6) to approximate the cardinality of the underlying set, by measuring the fraction of the bits that are "0." More specifically, the probability that a bit is *not* set is given below:

$$P(0) = \left(1 - \frac{1}{m}\right)^{nk}. \tag{12}$$

Solving for n gives us our estimate $\hat{n}$, which can be written as (13):

$$\hat{n} = \frac{\ln P(0)}{k \cdot \ln(1 - (1/m))}. \tag{13}$$

To summarize, given N unique RSUs, the transportation authority will estimate the underlying traffic flow as follows:

(1) With the cooperation of the trusted entities, decrypt the Bloom filters from the N receivers

(2) Compute (with a logical OR) the Bloom filters from all the set unions that appear in equation (10)

(3) Use equation (13) to estimate the cardinality of every set appearing in equation (10)

(4) Substitute the computed values into equation (10) and $|\cap_{i=1}^{N} A_i|$

Note that, equation (10) necessitates the enumeration of all combinations of i out of N elements, for $i = 1$ to N. As such, the computational complexity grows exponentially with N. Nevertheless, in our simulations, we were able to get reasonable running times for up to $N = 14$ receivers.

6. Simulation Experiments

In this section, we evaluate the performance of the proposed system in terms of accuracy and efficiency, and we compare our results with the current state-of-the-art approach. In addition, to test the accuracy of the system in a real environment, we run the protocol on a simulated road network model.

6.1. Simulation Environment. We simulated our system with a maximum of 14 receivers (RSUs) on a desktop machine with sixteen 3.0 GHz CPU cores and 64 GB of memory. To simulate the limited computational capabilities of the vehicles and RSUs, we employed a single CPU core to perform their tasks, i.e., Bloom filter encryption and aggregation, respectively. On the other hand, the server process that performs Bloom filter decryption and flow estimation utilized all sixteen cores in a multi-threaded implementation. Our code is written in C++ and we leveraged the OpenSSL library (https://www.openssl.org/) for arbitrary precision arithmetic operations. For sufficient security, the RSA modulus of the Paillier cryptosystem was set to 2048 bits, which produces ciphertexts of size 512 bytes.

6.2. Accuracy. Our system requires each RSU to maintain a Bloom filter of size m, containing n entries (vehicles). Since the main motive is to design a system for fine-grained traffic estimation, we considered a small number of vehicles (i.e., $n \leq 2000$) at each receiver. In this scenario, new Bloom filters are generated at relatively short time intervals. Each passing vehicle v sets k random bits at positions $H_1(v)$, $H_2(v)$, ..., $H_k(v)$ and, when n reaches a certain threshold (or a timer expires), the Bloom filter is sent to the transportation authority. A larger k results in more privacy (more uncertainty for an adversary) but incurs higher cost, because it requires larger Bloom filters. Specifically, we observed that the traffic flow estimation is more accurate when $m \geq kn$. However, the system assures sufficient privacy to vehicles with $k \geq 4$, as evident in equation (9). Similarly, the size of q also affects the accuracy of the system, because it is inversely proportional to the bit error probability P_{err}, as shown in equation (7). Nevertheless, our results indicated that the effect is negligible, even for a value as low as 128. To illustrate the simplicity and accuracy of our method, we used the same parameter settings in all the experiments: $k = 4$, $m = 8000$, and $q = 128$.

In our first set of experiments, we tested the traffic flow estimation accuracy for a varying number of receivers N (path length). We set the number of vehicles passing through a receiver to lie in the interval $[n_l, n_h]$ and varied the number of common vehicles across the N receivers in the range of 10%–75% of n_l (in increments of 1%). We also considered the case where the number of vehicles is identical across all receivers, i.e., $n_l = n_h$, which is common during peak hour traffic. As a performance metric, we used the average absolute difference (AAD) between the real and estimated traffic flows. In particular, we performed each experiment 1000 times and computed the AAD as follows:

$$\text{AAD} = \frac{\sum_{i=1}^{1000} |n_i - \hat{n}_i|}{1000}, \tag{14}$$

where n_i and $\hat{n}_i$ are the actual and estimated traffic flows at iteration i.

Figure 2 shows the AAD (in percentage w.r.t the real traffic flow) as a function of the real traffic flow, for the case where the number of vehicles at each RSU is constant ($n_l = n_h = n$). One important observation is that the accuracy does not decline when estimating the traffic flow across multiple RSUs; rather, the accuracy is improved significantly when involving more RSUs in the estimation. The second observation is that the estimation accuracy improves with increasing traffic flow, i.e., when the Bloom filters share a large number of common vehicles. For example, for 10 RSUs, $n = 2000$, and a real traffic flow of 200, the AAD is equal to 15, i.e., 7.5% of the real traffic flow. On the other hand, when the real traffic flow rises to 1500 vehicles, the AAD is just 12 (or 0.8%).

Nevertheless, in most cases, the number of vehicles captured by each RSU will vary. Therefore, in the second set of experiments, we measured the accuracy under this more realistic scenario. Figure 3 illustrates the results, which are quite similar to the case where n is fixed. In particular, we observe a minor impact on the estimation accuracy under longer path lengths, especially when the real traffic flow is low.

6.3. Comparison against the State-of-the-Art Protocol. In our earlier work [14], we compared our point-to-point traffic flow estimation protocol against Zhou et al.'s [13] method that is designed specifically for origin-destination flows. Therefore, in this study, we focus our comparison on Zhou et al.'s multipoint traffic estimation protocol [12] that can handle paths of arbitrary length. Recall that their method is similar to ours, in that they utilize a bit array of size m to encode the vehicle IDs that pass through an RSU. However, instead of a Bloom filter with k hash functions, they require each vehicle to submit one of s precomputed indices (chosen randomly) to each encountered RSU. In addition to the AAD, we also computed the standard deviation (σ) of the estimated values from the actual traffic, based on the following formula:

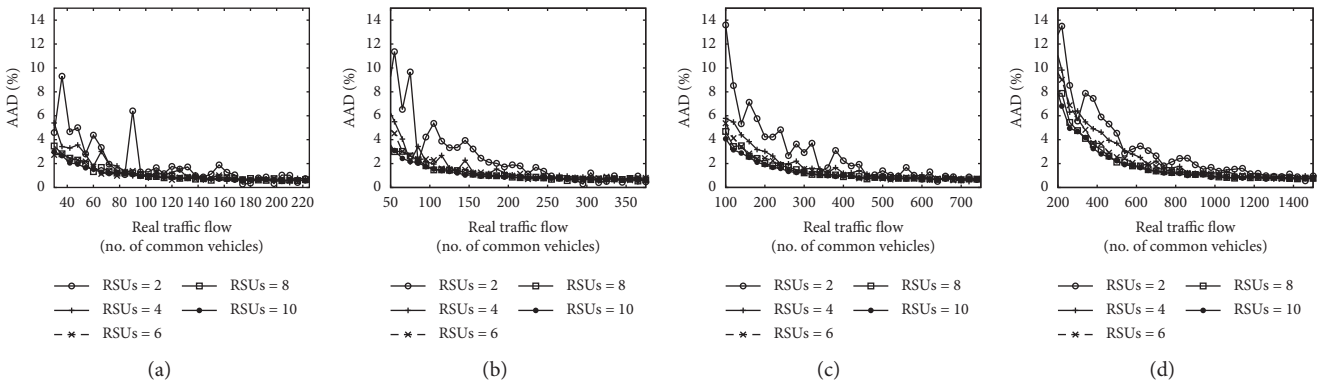

Figure 2: AAD vs. real traffic flow (fixed n across all RSUs). (a) $n = 300$. (b) $n = 500$. (c) $n = 1000$. (d) $n = 2000$.

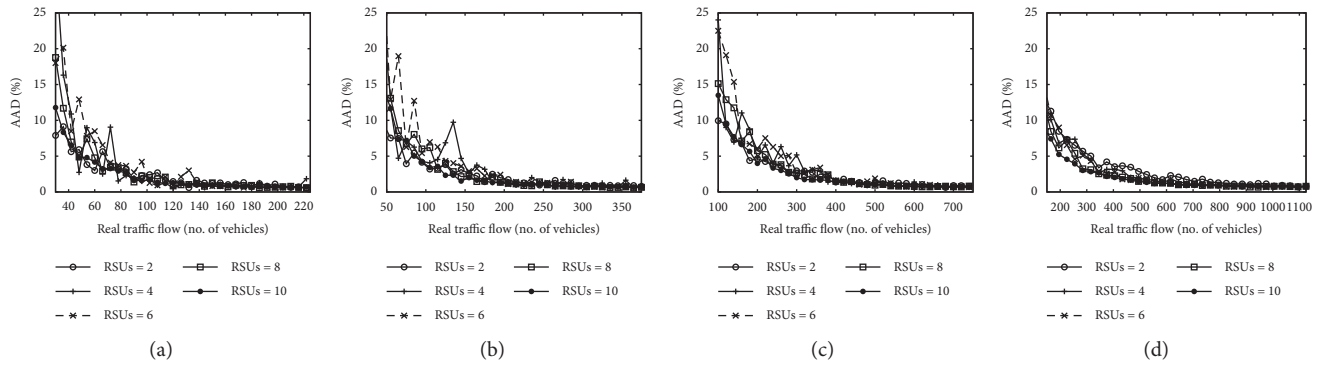

Figure 3: AAD vs. real traffic flow (variable n at each RSU). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

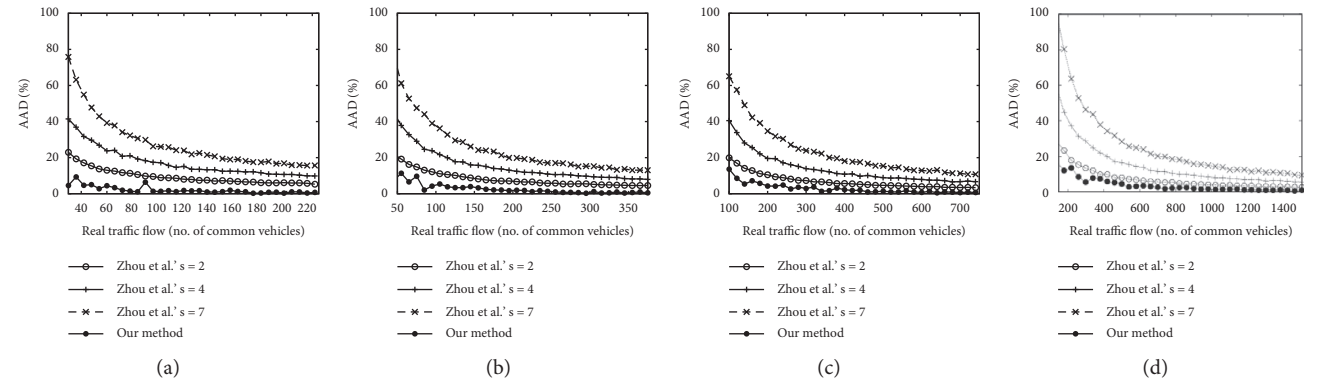

Figure 4: Performance comparison with respect to AAD for $N = 2$ RSUs (fixed n across all RSUs, $m = 8000$). (a) $n = 300$. (b) $n = 500$. (c) $n = 1000$. (d) $n = 2000$.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{1000} \left(n_i - \widehat{n}_i\right)^2}{1000}}. \tag{15}$$

Again, n_i and $\widehat{n}_i$ are the actual and estimated traffic flows at iteration i.

In the first experiment, we consider a fixed number n of vehicles at each RSU, and also use the same bit array size $m = 8000$ for both methods. For Zhou et al., we tested three different versions, namely, for $s \in \{2, 4, 7\}$. Figures 4 and 5 depict the AAD (%) and standard deviation, respectively, as a function of the real traffic flow (for $N = 2$ receivers). Our approach clearly outperforms the competitors, especially for larger values of s. When $s \geq 4$, our method reduces the AAD by a factor of 3–15. Note that, the value $s = 2$ is not considered privacy-preserving, because it is very easy for an adversary to track the same vehicle across different RSUs (a vehicle can only choose between two random indices). Figures 6 and 7 show the same experiments for $N = 3$ receivers. Our approach is clearly superior to the state-of-the-art protocol, and the performance gap is increased significantly compared to the case of the 2 receivers. We do not present results for $N > 3$ receivers, because the accuracy of Zhou et al.'s

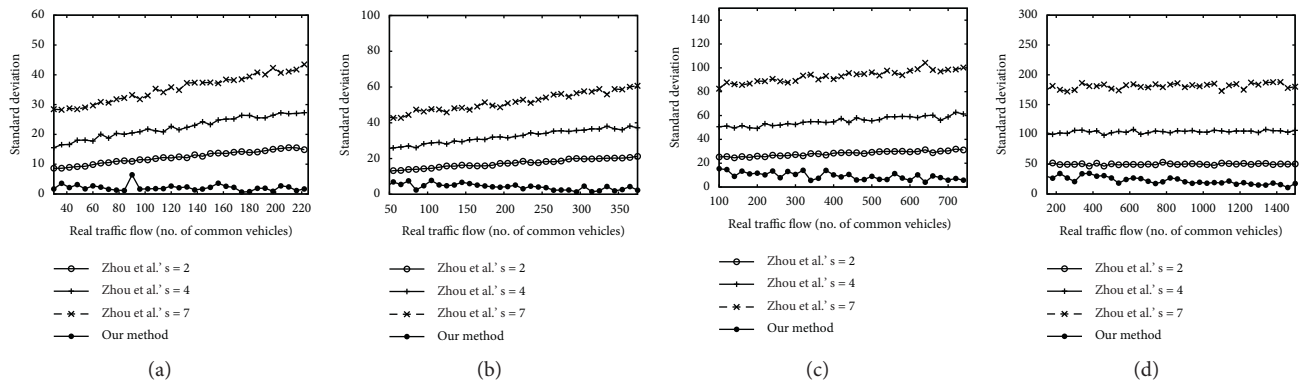

Figure 5: Performance comparison with respect to standard deviation for $N = 2$ RSUs (fixed n across all RSUs, $m = 8000$). (a) $n = 300$. (b) $n = 500$. (c) $n = 1000$. (d) $n = 2000$.

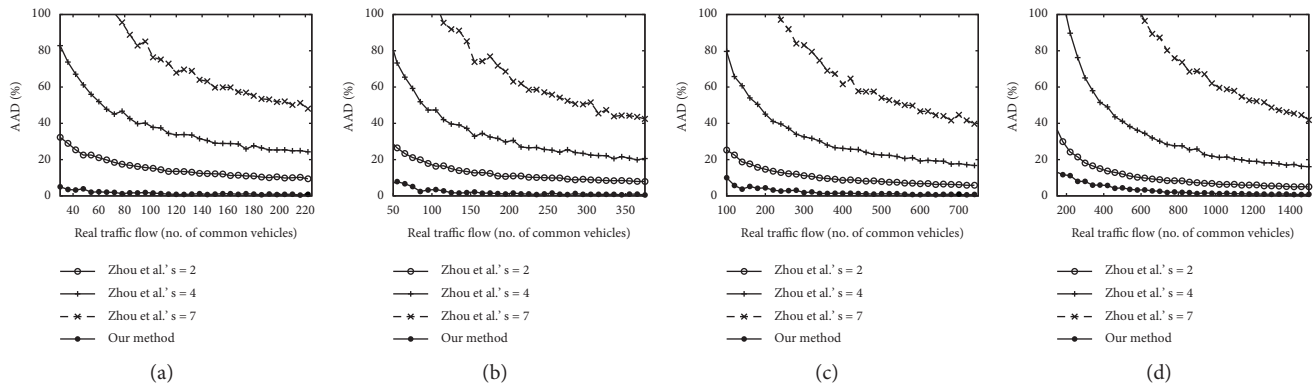

Figure 6: Performance comparison with respect to AAD for $N = 3$ RSUs (fixed n across all RSUs, $m = 8000$). (a) $n = 300$. (b) $n = 500$. (c) $n = 1000$. (d) $n = 2000$.

approach drops considerably (this was also observed by the authors of [12]). On the other hand, our method is very accurate when the number of RSUs increases, as illustrated in the previous section.

Next, we consider the case where the number of vehicles n varies across the different RSUs. Figures 8 and 9 illustrate the AAD (%) and standard deviation, respectively, as a function of the real traffic flow ($N = 2$). Similarly, Figures 10 and 11 plot same performance metrics for the case of $N = 3$ receivers. Here, our protocol outperforms the state-of-the-art method by several orders of magnitude, especially for the case of 3 receivers.

To this end, we also tested the effect of m on Zhou et al.'s accuracy. In particular, the authors proposed to configure the bit array size at each RSU as $m = 2^{\log(nf)}$, where n is the number of vehicles in the current measurement period and f is the load factor. Following the authors' recommendations, we set $f = 4$ and repeated the experiment where the number of vehicles n varies across the RSUs. The results are depicted in Figures 12–15. Overall, there is not any significant improvement in terms of AAD and standard deviation. In fact, the accuracy is negatively affected when the variance of n across the RSUs is small (i.e., when $n \in [1500, 2000]$). Our protocol consistently outperforms the competitors by a very large factor and, more importantly, it does not exhibit any loss in accuracy when increasing the path length of the measurement. This is a very desirable feature for intelligent transportation systems that need to estimate the traffic flow on specific paths along the road network, instead of on an origin-destination basis.

6.4. Overhead. The main advantage of the current state-of-the-art protocols is the lack of cryptographic operations that results in a very efficient implementation. However, this has a negative impact on both the privacy of the vehicles and the accuracy of the traffic flow estimation. Therefore, to illustrate the feasibility of our approach, we need to investigate the overhead of the cryptographic operations in terms of both computation and communication costs. These costs are directly related to the size of the encrypted Bloom filter, which is a function of the chosen parameters n, m, and q.

6.4.1. Computational Overhead. This is the processing time cost that relates to the cryptographic operations at each involved entity, i.e., the vehicles, the RSUs, and the server. To this end, the two basic operations involved in our methods are the modular exponentiation (for Paillier encryption/decryption) and the modular multiplication (for Paillier homomorphic addition). In our software implementation on a single-core processor, these operations cost, on average, 8 ms and 0.015 ms, respectively. Notice that the overhead of

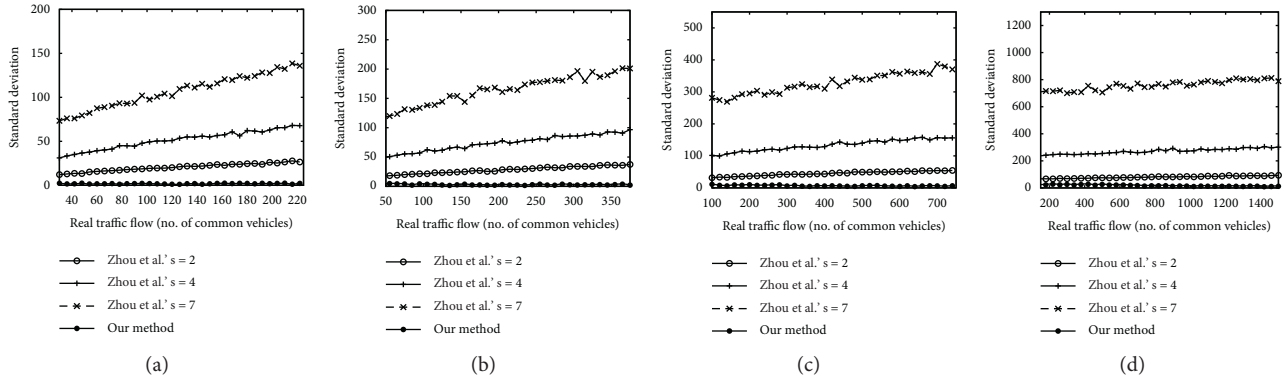

Figure 7: Performance comparison with respect to standard deviation for $N = 3$ RSUs (fixed n across all RSUs, $m = 8000$). (a) $n = 300$. (b) $n = 500$. (c) $n = 1000$. (d) $n = 2000$.

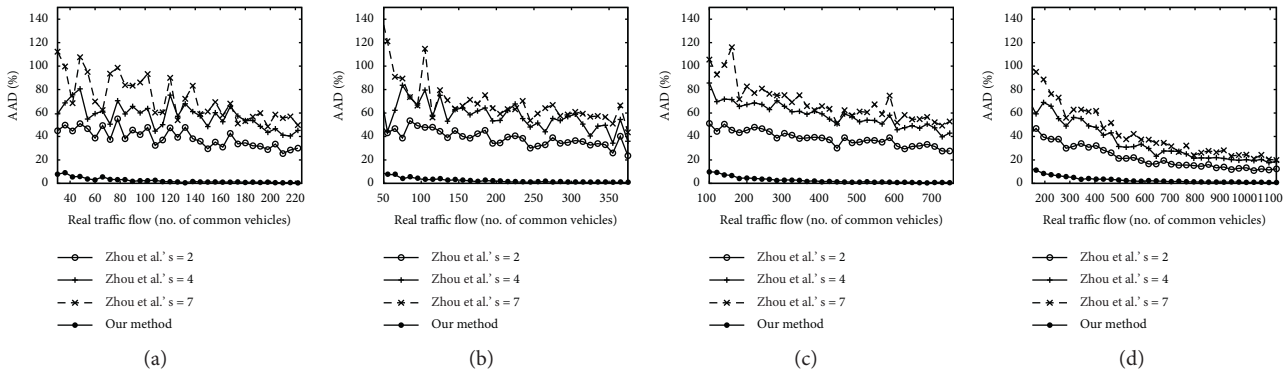

Figure 8: Performance comparison with respect to AAD for $N = 2$ RSUs (variable n at each RSU, $m = 8000$). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

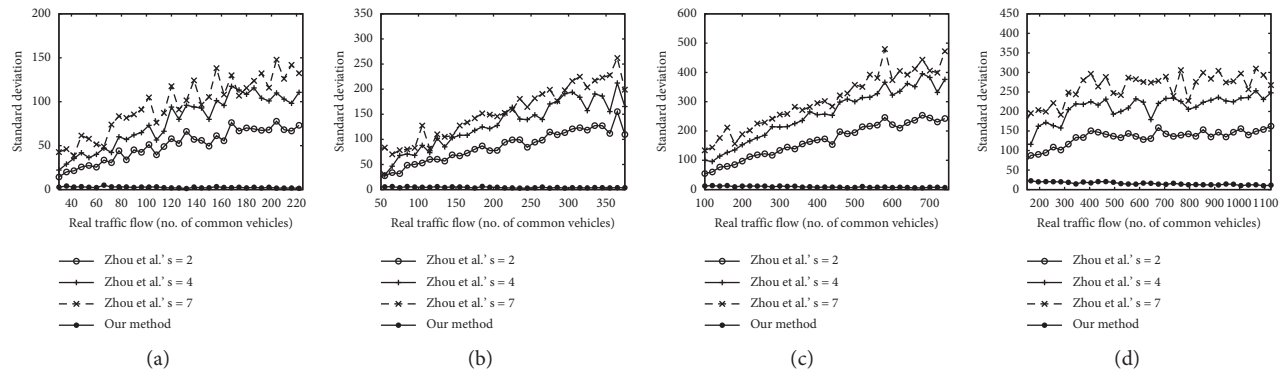

Figure 9: Performance comparison with respect to standard deviation for $N = 2$ RSUs (variable n at each RSU, $m = 8000$). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

the OTP operations is negligible compared to the public key operations and is, thus, not measured in our results.

Figure 16 shows the computational cost at the vehicle and the server as a function of the bit-length of q. For the vehicle, the cost involves the encryption of the OTP keys into multiple Paillier ciphertexts. The processing time grows slowly with increasing bit-length, because more ciphertexts are needed to store the entire Bloom filter. Even in the worst-case configuration, when $q = 2^{16}$ and $m = 16000$, the cost is just 1.8 sec, which is long enough for the vehicle to compute a new Bloom filter before reaching the next RSU. It is also possible for the vehicle to precompute (offline) several Bloom filters before the start of a new trip.

Similarly, the cost at the server shows the time needed to decrypt a single aggregate Bloom filter (each trusted party will incur this cost). Here, the server application employs all

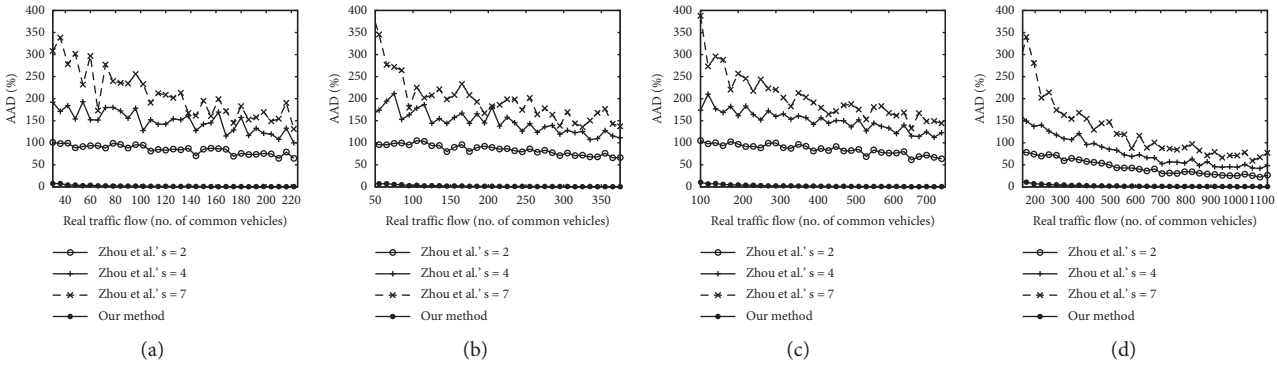

Figure 10: Performance comparison with respect to AAD for $N = 3$ RSUs (variable n at each RSU, $m = 8000$). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

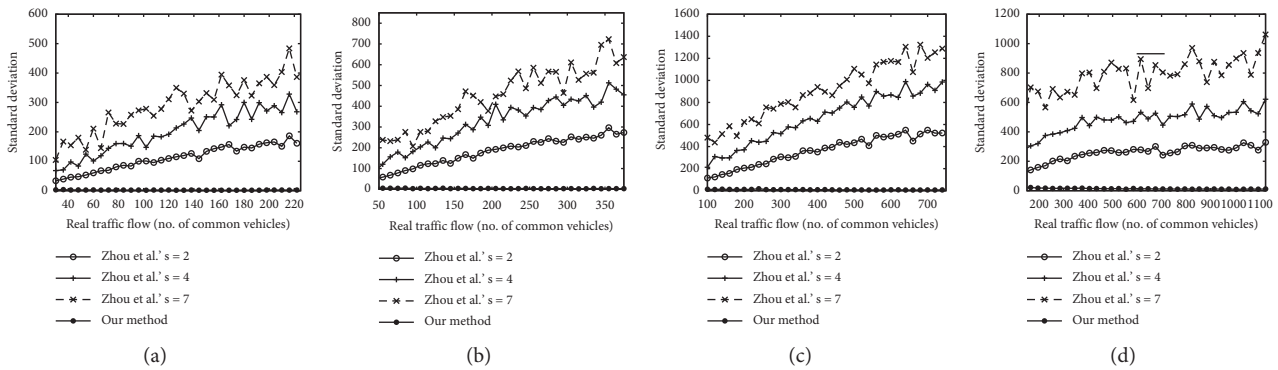

Figure 11: Performance comparison with respect to standard deviation for $N = 3$ RSUs (variable n at each RSU, $m = 8000$). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

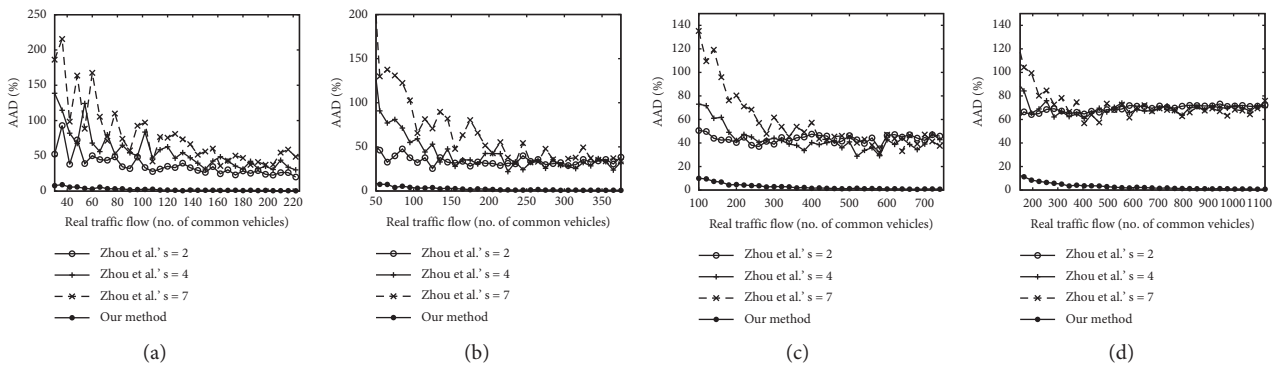

Figure 12: Performance comparison with respect to AAD for $N = 2$ RSUs (variable n at each RSU, $m = 8000$ for our method, and $m = 2^{\log(nf)}$ for Zhou et al.). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

sixteen CPU cores, so the cost is greatly reduced. With the worst-case configuration, i.e., $q = 2^{16}$ and $m = 16000$, the decryption cost for one Bloom filter is just 310 ms. On the other hand, Figure 17 depicts the processing time required at the server to estimate the traffic flow across N receivers, using equation (10). As we explained in Section 5, the cost grows exponentially with N, because it necessitates the enumeration of all combinations of i out of N elements, for $i = 1$ to N. This cost clearly limits the maximum path length N that can be supported, but Figure 17 shows that the cost is quite reasonable for $N < 15$.

Finally, the computational cost at the RSU involves only modular multiplications, which are considerably cheaper than exponentiations. In our implementation of the RSU-related operations, the RSU entails just 1 ms of CPU time to add one vehicle to the aggregate Bloom filter (with the configuration of $n = 2000$, $m = 8000$, and $q = 128$). At this rate, the RSU can process approximately 1000 vehicles/sec.

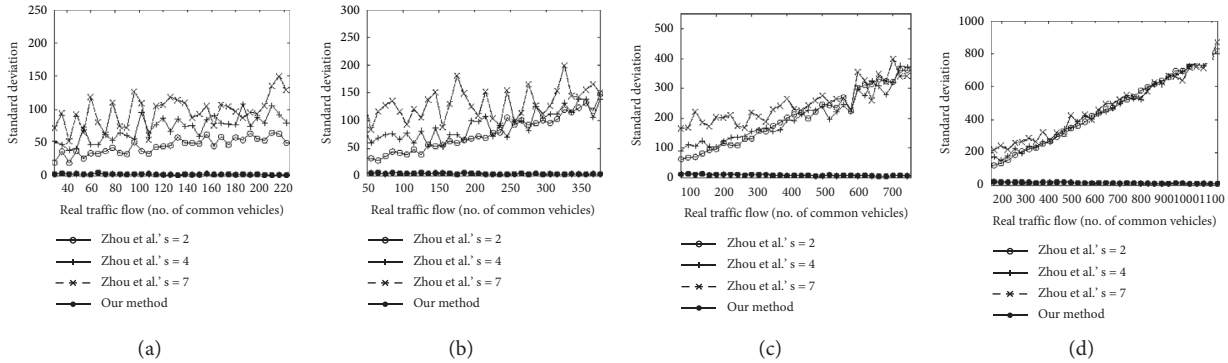

Figure 13: Performance comparison with respect to standard deviation for N = 2 RSUs (variable (n) at each RSU, m = 8000 for our method, and $m = 2^{\log(nf)}$ for Zhou et al.). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

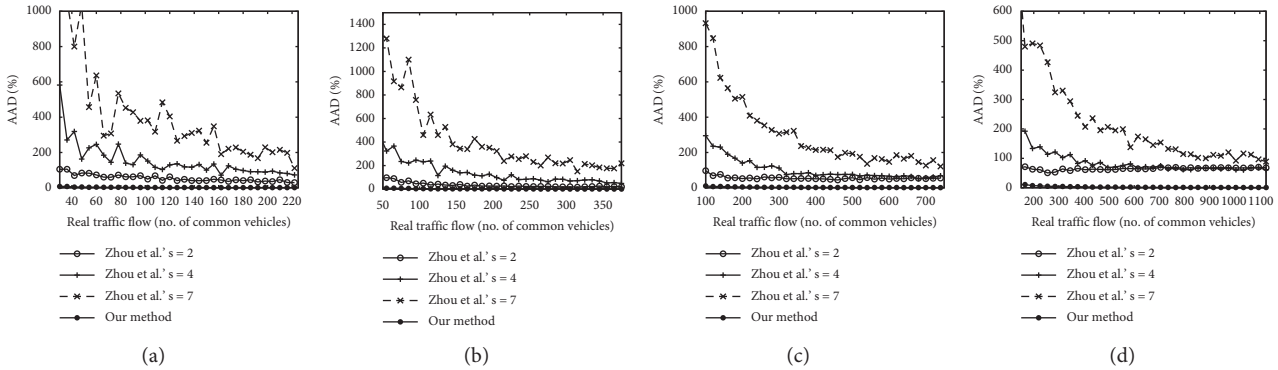

Figure 14: Performance comparison with respect to AAD for $N = 3$ RSUs (variable n at each RSU, $m = 8000$ for our method, and $m = 2^{\log(nf)}$ for Zhou et al.). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

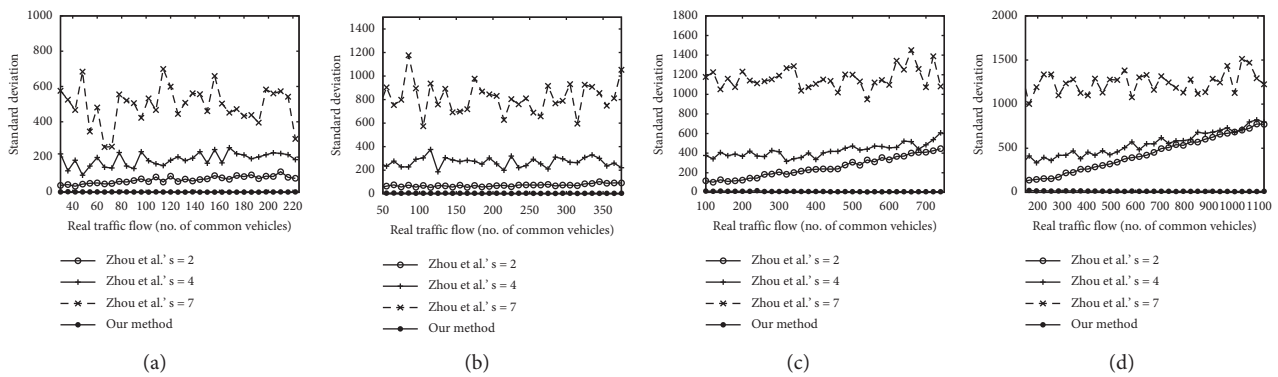

Figure 15: Performance comparison with respect to standard deviation for $N = 3$ RSUs (variable n at each RSU, $m = 8000$ for our method, and $m = 2^{\log(nf)}$ for Zhou et al.). (a) $n \in [300, 600]$. (b) $n \in [500, 1000]$. (c) $n \in [1000, 2000]$. (d) $n \in [1500, 2000]$.

6.4.2. Communication Overhead. The communication cost is measured as the bandwidth usage between (i) the vehicles and the RSUs, and (ii) the RSUs and the server. It is worth noting that the communication cost is the bottleneck with regard to the vehicle processing throughput at the RSU. Indeed, the data rate of the DSRC protocol is between 6 and 27 Mbps, which can only support a limited number of Bloom filter transmissions within any given time period. As an example, when $n = 2000$, $m = 8000$, and $q = 128$, the size of a single Bloom filter is just 43 KB. Figure 18 shows the processing throughput as a function of the available bandwidth, which demonstrates that at 10 Mbps, the system is able to accommodate the load of a typical rush hour traffic.

On the other hand, the bandwidth usage between the RSUs and the server is considerably less. Specifically, after each aggregation period (which is in the order of a few

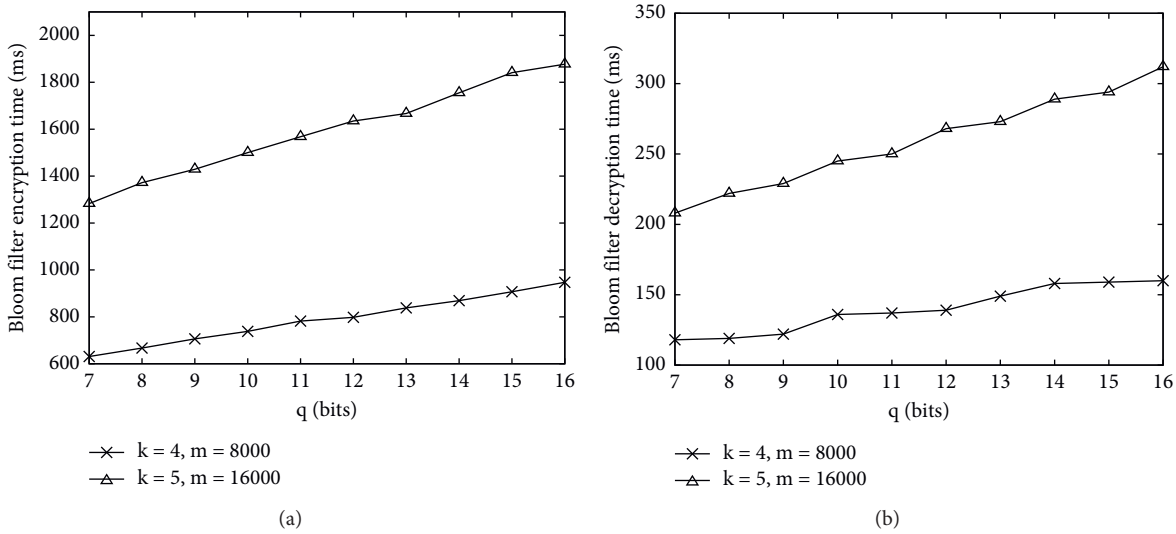

Figure 16: Computational cost of Bloom filter encryption (at vehicle) and decryption (at server) ($n = 2000$). (a) Vehicle. (b) Server.

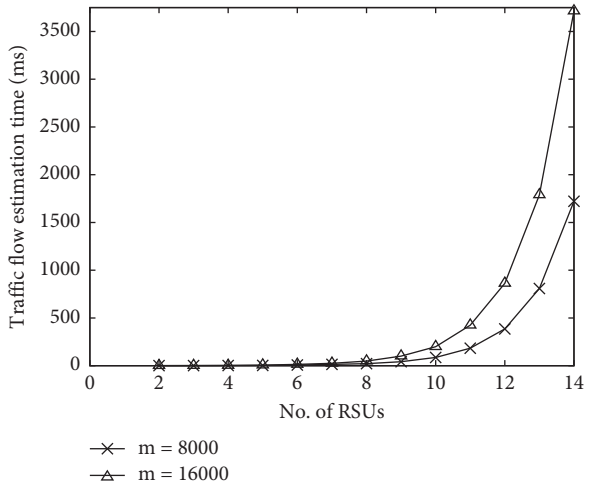

Figure 17: Traffic flow estimation time vs. number of RSUs.

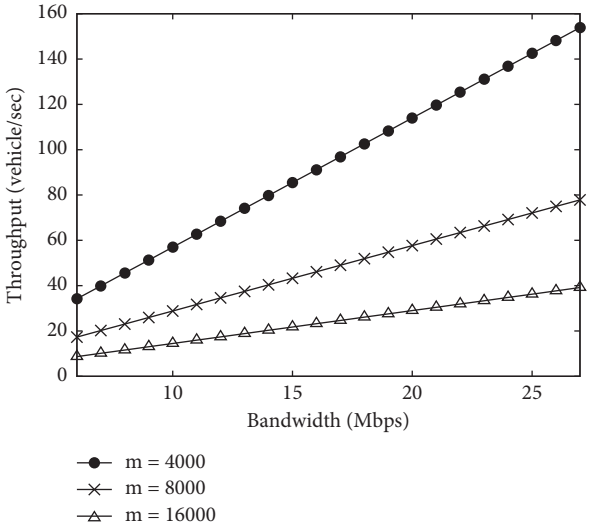

Figure 18: Throughput vs. DSRC bandwidth ($n = 2000$, $q = 128$).

minutes), the RSU has to send to the server a *single* aggregate Bloom filter. In our example above, this entails just 43 KB of data, e.g., an average data rate of 144 bytes/sec, for an aggregation period of 5 min. This is a data rate that is easily supported by 3G communication technologies.

6.4.3. Road Network Simulation. To test the accuracy of our system in a more realistic environment, we estimated the traffic flows on a simulated traffic model, using a road segment with multiple entry and exit points for vehicles. The model is depicted in Figure 19. In particular, we consider five RSUs deployed along the road segment, where each consecutive pair is separated by a distance of 5 km (i.e., the length of the road segment is 20 km). There are a total of 2000 vehicles, with their entry-exit points fixed, as shown in the figure. For simplicity, all vehicles start entering the road segment at time $t_0 = 0$, at a rate of one vehicle per 60 ms. Furthermore, we assume that every vehicle moves at a speed that is uniformly distributed in the interval 60–80 km/h. As such, the time needed to pass through two consecutive RSUs is upper bounded by 5 min, and the total simulation time is just over 20 min. Finally, we assume that each RSU generates a new Bloom filter every 5 min, so, to compute the traffic flow for a given time window (t_i, t_j), the server has to aggregate (for each involved RSU) the Bloom filters that fall within this time window. The aggregation is done with a logical OR of the individual Bloom filters, and the traffic flow among the selected RSUs is estimated via equation (10). The source code of our simulation is available on GitHub (https://github.com/rathoremazhar/Traffic-Flow-Estimation).

Figure 20 illustrates the real and estimated traffic flows for different combinations of RSUs. Specifically, each arrow indicates the involved RSUs, while the time window T is depicted on the left-hand side of the figure. The number above each arrow conveys the estimated value and the one below shows the real traffic flow, i.e., the number of common vehicles. In our simulation, a vehicle needs at most five

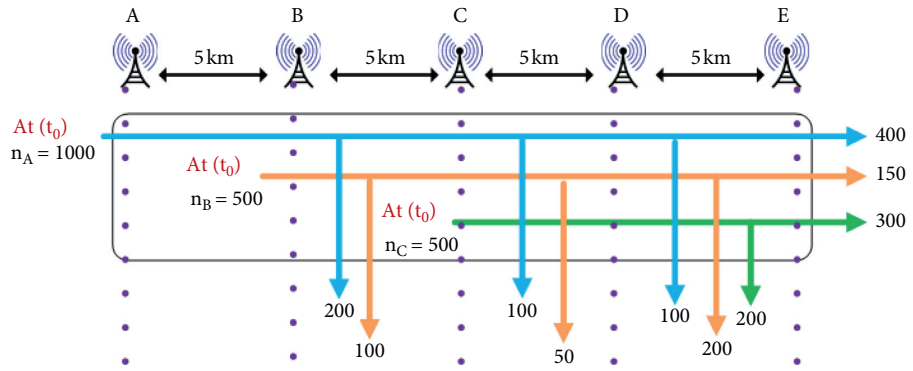

Figure 19: The simulated traffic flow model on a road network.

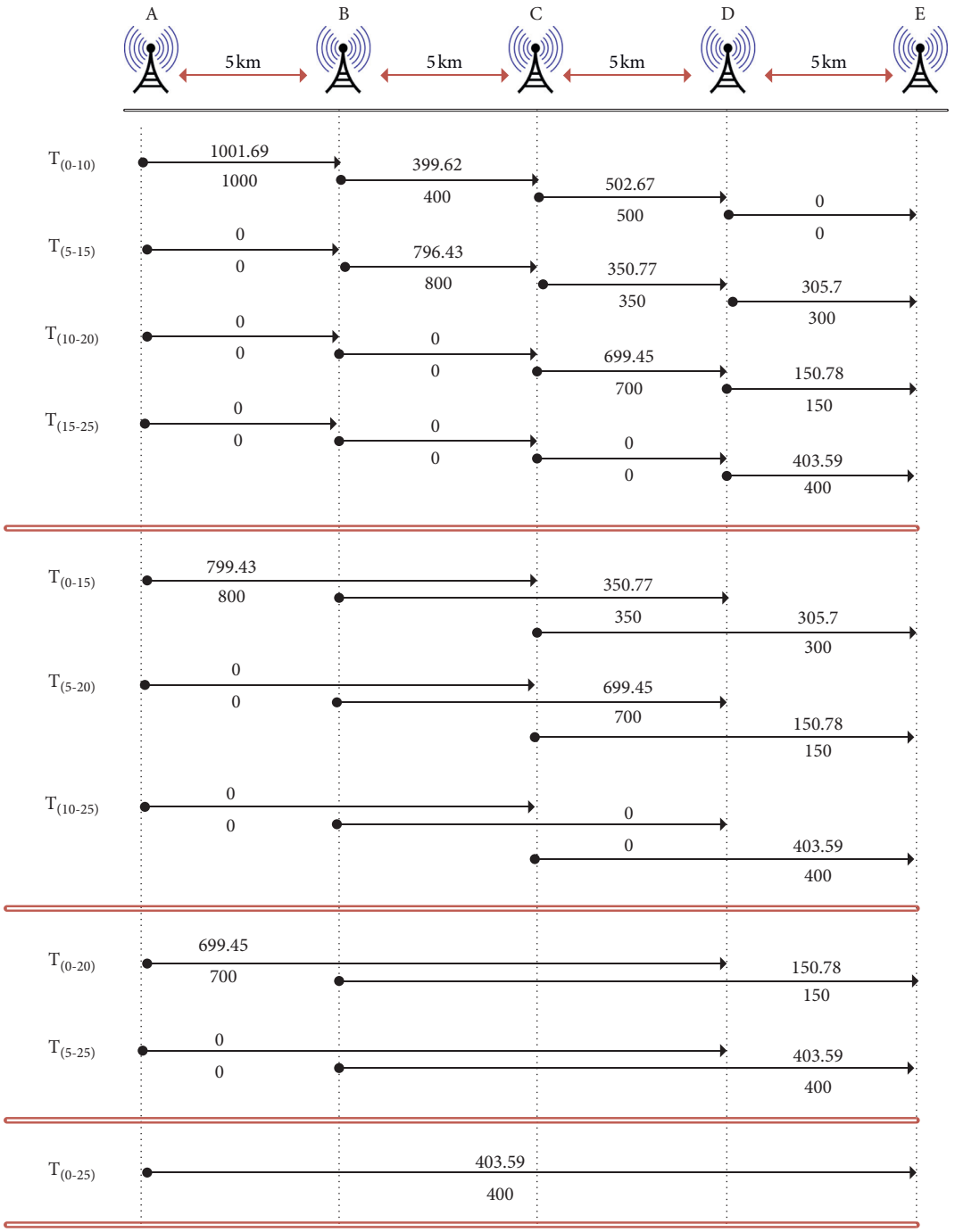

Figure 20: Estimated vs. real traffic flows for the simulated model.

minutes to travel from one RSU to the next, so, to accurately compute the traffic flow between N RSUs, we must aggregate the Bloom filters from a time window of at least $5N$ minutes. For instance, to estimate the traffic flow between receivers A, B, and C, the correct time window to choose is (0, 15]. As evident in Figure 20, our protocol produces very accurate results, regardless of the number of RSUs or the underlying traffic volume.

7. Conclusions

We proposed a very simple and accurate protocol for estimating—in a privacy-preserving manner—the traffic flow across arbitrary paths on a road network. Our solution leverages roadside units that interact with the passing vehicles, in order to construct encrypted Bloom filters that summarize the underlying vehicle IDs. We performed an extensive simulation study, using diverse traffic characteristics, and our results are extremely promising. In particular, we demonstrated that our protocol's estimations exhibit only a minor deviation from the real traffic flow. More importantly, the accuracy is maintained regardless of the underlying path length. We also compared our approach with the current state-of-the-art protocols and showed that it improves the estimation accuracy by a large factor. Finally, we implemented the cryptographic primitives involved in our method and demonstrated the feasibility and scalability of the system.

Data Availability

No data were used to support this study.

Disclosure

This paper is an extension of the conference paper "Privacy-preserving traffic flow estimation for road networks," by E. Bentafat, M. M. Rathore, and S. Bakiras, published in Proc. IEEE GLOBECOM 2020 (https://ieeexplore.ieee.org/document/9322454).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] M. Castro-Neto, Y. Jeong, M. K. Jeong, and L. D. Han, "AADT prediction using support vector regression with data-dependent parameters," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2979–2986, 2009.

[2] J. K. Eom, M. S. Park, T.-Y. Heo, and L. F. Huntsinger, "Improving the prediction of annual average daily traffic for nonfreeway facilities by applying a spatial statistical method," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1968, no. 1, pp. 20–29, 2006.

[3] W. H. K. Lam and J. Xu, "Estimation of AADT from short period counts in Hong Kong - a comparison between neural network method and regression analysis," *Journal of Advanced Transportation*, vol. 34, no. 2, pp. 249–268, 2000.

[4] D. Mohamad, K. C. Sinha, T. Kuczek, and C. F. Scholer, "Annual average daily traffic prediction model for county roads," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1617, no. 1, pp. 69–77, 1998.

[5] Y. Lou and Y. Yin, "A decomposition scheme for estimating dynamic origin-destination flows on actuation-controlled signalized arterials," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 5, pp. 643–655, 2010.

[6] N. Caceres, J. P. Wideberg, and F. G. Benitez, "Deriving origin-destination data from a mobile phone network," *IET Intelligent Transport Systems*, vol. 1, no. 1, pp. 15–26, 2007.

[7] J. White and I. Wells, "Extracting origin destination information from mobile phone data," in *Proceedings of the International Conference on Road Transport Information and Control*, London, UK, March 2002.

[8] T. Jeske, "Floating car data from smartphones: what google and waze know about you and how hackers can control traffic," *Proceedings of the BlackHat Europe*, pp. 1–12, 2013.

[9] C. Nanthawichit, T. Nakatsuji, and H. Suzuki, "Application of probe-vehicle data for real-time traffic-state estimation and short-term travel-time prediction on a freeway," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1855, no. 1, pp. 49–59, 2003.

[10] Y. Zhu, Y. Wu, and B. Li, "Vehicular ad hoc networks and trajectorybased routing," in *Internet of Things*, pp. 143–167, Springer, Berlin, Germany, 2014.

[11] Y. L. Morgan, "Notes on DSRC & WAVE standards suite: its architecture, design, and characteristics," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 4, pp. 504–518, 2010.

[12] Y. Zhou, S. Chen, Y. Zhou, M. Chen, and Q. Xiao, "Privacy-preserving multi-point traffic volume measurement through vehicle-to-infrastructure communications," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5619–5630, 2015.

[13] Y. Zhou, Z. Mo, Q. Xiao, S. Chen, and Y. Yin, "Privacy-preserving transportation traffic measurement in intelligent cyber-physical road systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3749–3759, 2016.

[14] E. Bentafat, M. M. Rathore, and S. Bakiras, "Privacy-preserving traffic flow estimation for road networks," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Taipei, Taiwan, December 2020.

[15] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[16] I. Tsapakis, W. H. Schneider, and A. P. Nichols, "A Bayesian analysis of the effect of estimating annual average daily traffic for heavy-duty trucks using training and validation data-sets," *Transportation Planning and Technology*, vol. 36, no. 2, pp. 201–217, 2013.

[17] B. Yang, S.-G. Wang, and Y. Bao, "Efficient local AADT estimation via SCAD variable selection based on regression models," in *Proceedings of the Chinese Control and Decision Conference (CCDC)*, pp. 1898–1902, Mianyang, China, May 2011.

[18] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Enhancing security and privacy in traffic-monitoring systems," *IEEE Pervasive Computing*, vol. 5, no. 4, pp. 38–46, 2006.

[19] B. Hoh and M. Gruteser, "Protecting location privacy through path confusion," in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pp. 194–205, Athens, Greece, September 2005.

[20] A. Tomandl, D. Herrmann, and H. Federrath, "PADAVAN: privacyaware data accumulation for vehicular ad-hoc networks," in *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 487–493, Larnaca, Cyprus, October 2014.

[21] S. Rass, S. Fuchs, M. Schaffer, and K. Kyamakya, "How to protect privacy in floating car data systems," in *Proeedings of the ACM International Workshop on Vehicular Ad Hoc Networksd (VANET)*, pp. 17–22, San Francisco, CA, USA, 2008.

[22] B. Hoh, M. Gruteser, R. Herring et al., "Virtual trip lines for distributed privacy-preserving traffic monitoring," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pp. 15–28, Breckenridge, CO, USA, June 2008.

[23] D. Forster, H. L¨ ohr, and F. Kargl, "Decentralized enforcement of k-¨ anonymity for location privacy using secret sharing," in *Proceedings of the IEEE Vehicular Networking Conference (VNC)*, pp. 279–286, Kyoto, Japan, December 2015.

[24] Y. Zhou, S. Chen, Z. Mo, and Y. Yin, "Privacy preserving origindestination flow measurement in vehicular cyber-physical systems," in *Proceedings of the 2013 IEEE 1st International Conference on Cyber-Physical Systems, Networks, and Applications (CPSNA)*, pp. 32–37, Taipei, Taiwan, August 2013.

[25] Y.-E. Sun, H. Huang, S. Chen, H. Xu, K. Han, and Y. Zhou, "Persistent traffic measurement through vehicle-to-infrastructure communications in cyber-physical road systems," *IEEE Transactions on Mobile Computing*, vol. 18, no. 7, pp. 1616–1630, 2018.

[26] Y.-E. Sun, H. Huang, S. Chen, Y. Zhou, K. Han, and W. Yang, "Privacy-preserving estimation of k -persistent traffic in vehicular cyber-physical systems," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8296–8309, 2019.

[27] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: theory and implementation," *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, p. 79, 2018.

[28] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pp. 169–178, Bethesda, MD, USA, June 2009.

[29] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the International Conference on the Theory and applications of Cryptographic Techniques*, pp. 223–238, Springer, Prague, Czech Republic, May 1999.

[30] I. Damgard and M. Jurik, "A generalisation, a simplification and some˚ applications of paillier's probabilistic public-key system," *Proceedings of the International Workshop on Practice and Theory in Public Key Cryptography (PKC)*, vol. 1992, pp. 119–136, 2001.

[31] L. Fan, P. Cao, J. M. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, 2000.

[32] C. Dwork, "Differential privacy," in *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, pp. 1–12, Venice, Italy, July 2006.