Enhancing Privacy and Accuracy in Probe Vehicle Based Traffic Monitoring via Virtual Trip Lines

Baik Hoh∗, Toch Iwuchukwu∗, Quinn Jacobson∗
Daniel Work¶, Alexandre M. Bayen¶, Ryan Herring¶, Juan-Carlos Herrera¶
Marco Gruteser†, Murali Annavaram§, Jeff Ban‡

Abstract— **Traffic monitoring using probe vehicles with GPS receivers promises significant improvements in cost, coverage, and accuracy over dedicated infrastructure systems. Current approaches, however, raise privacy concerns because they require participants to reveal their positions to an external traffic monitoring server. To address this challenge, we describe a system based on virtual trip lines and an associated cloaking technique, followed by another system design in which we relax the privacy requirements to maximize the accuracy of real-time traffic estimation.**

We introduce virtual trip lines which are geographic markers that indicate where vehicles should provide speed updates. These markers are placed to avoid specific privacy sensitive locations. They also allow aggregating and cloaking several location updates based on trip line identifiers, without knowing the actual geographic locations of these trip lines. Thus, they facilitate the design of a distributed architecture, in which no single entity has a complete knowledge of probe identities and fine-grained location information. We have implemented the system with GPS smartphone clients and conducted a controlled experiment with 100 phone-equipped drivers circling a highway segment, which was later extended into a year-long public deployment.

I. Introduction

Personal navigation services in vehicles enable the effective delivery and presentation of high resolution traffic information to drivers. At the same time, there is an increased need for data collection on currently unmonitored roadways, and traffic estimation algorithms to process this data. Traditionally, traffic data collection mechanisms have relied on fixed sensor networks, including inductive loop detectors, wireless magnetometer sensors, and microwave radar sensors. Because these dedicated sensing systems are expensive to install and maintain, their deployment has been limited largely to highways. As a result, traffic information on many of the major arterial roads is sorely lacking.

GPS probe vehicle based systems promise to significantly improve coverage and timeliness of traffic information [6], [7], [8]. Systems relying on probe data estimate traffic conditions with GPS measurements fused with traditional sources of traffic information such as loop detectors, camera, and human reports. With sufficient penetration (fraction of total traffic) this approach could potentially enable the collection of real-time traffic information over the complete road network at minimal cost for transportation agencies.

Several studies have demonstrated the feasibility of probe based traffic estimation through analysis, simulations, and experiments [12], [14], [28]. Yet several challenges must be addressed for successful deployments. First, a probe based system requires that cars reveal their positions to a traffic monitoring organization, raising privacy concerns. Hoh et al. [27] have proposed privacy enhancing technologies that can alleviate concerns. These solutions, however, still require users to trust centralized privacy servers. In addition, the system must be bootstrapped with other sources of information, since accurate estimates can only be achieved when sufficient users participate. While it remains possible to leverage existing telematics platforms or navigation systems, these platforms are not openly programmable and thus hard to retrofit for this purpose. Third, the collected data for the system should be as accurate as possible. The accuracy of data could be degraded by GPS positioning inaccuracy and bogus data injected by malicious users.

To address these challenges, we propose a novel traffic monitoring system design based on the concept of *virtual trip lines (VTLs)* and experimentally evaluate its feasibility. Virtual trip lines are geographic markers stored in the mobile phone client, which trigger a position and speed update when a probe vehicle trajectory intersects a trip line. Through privacy-aware placement of these trip lines, clients need not rely on a trustworthy server. The system is designed for GPS-enabled cell phones to enable rapid software deployment to a large and increasing number of programmable smart phones. As an extended version of our earlier paper [25], key contributions of this work include:

- Arguing that spatial sampling (through virtual trip lines) rather than temporal sampling leads to increased privacy because it allows omitting location samples from more sensitive areas.
- Describing a privacy-aware placement approach that creates the virtual trip line database.
- Demonstrating that the virtual trip line concept can be implemented on a GPS-enabled cellular phone platform.
- Evaluating accuracy and privacy through more extensive datasets from a large-scale field experiment (Mobile Century [22]) in the San Francisco Bay Area.
- Developing a light-weight trip line crossing detection algorithm against inaccurate GPS readings and intermittent

∗Nokia Research Center, CA USA
¶Dept. of Civil and Environmental Eng., UC Berkeley, CA USA
†Dept. of Electrical and Computer Eng., Rutgers Univ., NJ USA
§Dept. of Electrical Eng., Univ. of Southern California, CA USA
‡Dept. of Civil and Environmental Eng., Rensselaer Polytechnic Institute, NY USA

wireless connectivity.

The remainder of this article is organized as follows. Section II describes the challenges in probe vehicle based traffic monitoring. Section III introduces the virtual trip line concept and discusses its potential uses in the domain of traffic monitoring. Section IV describes the use of VTLs in two different traffic monitoring architectures and discusses the privacy features of each system. We implement and evaluate proposed architectures in section V and VI. Then we discuss limitations and outlooks in section VII, and propose some conclusions.

II. Traffic Monitoring Challenges in Probe Vehicle Systems

In this section we describe two challenges faced by probe monitoring systems, and our design goals to overcome these challenges through the implementation of traffic monitoring with virtual trip lines.

A. Privacy Risks

Traffic monitoring using GPS-equipped vehicles raises significant privacy concerns, because the external traffic monitoring entity acquires fine-grained movement traces of the probe vehicle drivers. These location traces might reveal sensitive places that drivers have visited, from which, for example, medical conditions, political affiliations, traffic violations, or potential involvement in traffic accidents could be inferred.

Threat Model and Assumptions. This work assumes that adversaries can compromise any single infrastructure component to extract information and can eavesdrop on network communications. We assume that different infrastructure parties do not collude. We believe that this model is useful in light of the many data breaches that occur due to dishonest insiders, hacked servers, stolen computers, or lost storage media (see [4] for an extensive list, including a dishonest insider case that released 4500 records from California's FasTrak automated road toll collection system). These cases usually involve compromised log files or databases in a single system component and motivate our approach of ensuring that no single infrastructure component can accumulate sensitive information.

We assume that a handset (i.e., a client application) itself is trustworthy but its owner can be malicious. Thus an owner cannot reverse-engineer the client code, so that he or she cannot intentionally manipulate a GPS reading, speed, timestamp of measurements, or cryptographic keys. However, as we will consider in Section VII-A, an owner can use the client application for malicious purposes within the legitimate use of a handset. We call this situation *compromised phones*. For example, a company competing for the same service (e.g., traffic monitoring services) can hire multiple users and ask them to intentionally drive slow in non-congested roads.

We label sensitive information any information from which the precise location of an individual at a given time can be inferred. Traffic monitoring does not need to rely on individuals or personal information, only on the aggregated statistics from a large number of probe vehicles. Thus, an obvious privacy measure is to anonymize the location data by removing identifiers such as network addresses. This approach is insufficient, however, because drivers can often be reidentified by correlating anonymous location traces with identified data from other sources. For example, home locations can be identified from anonymous GPS traces [26], [31] which may be correlated with address databases to infer the likely driver. Similarly, records on work locations or automatic toll booth records could help identify drivers. Even if anonymous point location samples from several drivers are mixed, it is possible to reconstruct individual traces because successive location updates from the same vehicle inherently share a high spatio-temporal correlation. If overall probe vehicle density is low, location updates close in time and space likely originate from the same vehicle. This approach is formalized in target tracking models [36].

As an example of tracking anonymous updates, consider the following problem: given a time series of anonymous location and speed samples mixed from multiple users, extract a subset of samples generated by the same vehicle. To this end, an adversary can predict the next location update ($\hat{x}_{t+\Delta t}$) based on the prior reported speed $\hat{x}_{t+\Delta t} = v_t \cdot \Delta t + x_t$ of the actual reported updates, where x_t and $x_{t+\Delta t}$ are locations at time t and $t + \Delta t$, respectively, and v_t is the reported speed at t. The adversary then associates the prior location update with the next update closest to the prediction, or more formally with the most likely update, where likelihood can be described through a conditional probability $P(x_{t+1}|x_t)$ that primarily depends on spatial and temporal proximity to the prediction. The probability can be modeled through a probability density function of distance (or time) differences between the predicted update and an actual update (under the assumption that the distance difference is independent of the given location sample).

Privacy Metrics. As observed in [27], the degree of privacy risk depends on how long an adversary successfully tracks a vehicle. Longer tracking increases the likelihood that an adversary can identify a vehicle and observe it visiting sensitive places. We thus adopt the *time-to-confusion* [27] metric and its variant *distance-to-confusion*, which measures the time or distance over which tracking may be possible. Distance-to-confusion is defined as the travel distance until tracking uncertainty rises above a defined threshold. Tracking uncertainty is calculated separately for each location update in a trace as the entropy $H = -\sum p_i \log p_i$, where the p_i are the normalized probabilities derived from the likelihood values described in [21]. These likelihood values are calculated for every location update generated within a temporal and spatial window after the location update under consideration.

These tracking risks and the observations regarding increased risks at certain locations further motivate the virtual trip line solution described next. Compared to a periodic update approach, in which clients provide location and speed updates at regular time intervals, virtual trip lines can be placed in a way to avoid updates from sensitive areas.

Goal. We aim to achieve privacy protection by design so that the compromise of a single entity, even by an insider at the service provider, does not allow individual users to be tracked or reidentified.

B. Lack of Guaranteed Accuracy of Sensor Data

The quality of traffic monitoring is contingent on the accuracy of the sensor data. In turn, the accuracy of this data is affected by technical limitations of sensor and the potential for maliciously injected bogus data. Thus, a key strategy to provide high quality traffic monitoring is to ensure accurate speed and location measurements in the presence of GPS error and to prevent malicious injection attacks.

To address the issue of GPS position errors, some level of client–side or server–side data filtering is required. If a light–weight algorithm running on the client can manage this job efficiently, it not only reduces user privacy concerns by avoiding data transmission, but also reduces the server–side computational burden, thereby achieving better scalability. To prevent bogus measurements from entering the data stream, some security countermeasures can be introduced to validate data authenticity. However, device authentication conflicts with user anonymity desired for privacy, and authentication alone cannot prevent fraudulent updates. Recent studies have presented a trusted platform module (TPM) [18], [38] for preventing fraudulent updates.

Goal. The client software must cope with the resource constraints of current cellphone platforms where the use of computationally expensive algorithms such as map-matching and Kalman filtering is limited. We mainly focus on designing a light-weight component that detects trip line crossings accurately while suppressing false positives in the presence of noisy GPS readings and intermittent wireless connectivity (which affects A-GPS performance). Additionally the system should not allow adversaries to insert spoofed data, which would compromise the data quality and thus traffic information. This is especially challenging because it conflicts with the desire for anonymity.

III. Virtual Trip Lines

To address these challenges our proposed traffic monitoring system builds on the novel concept of virtual trip lines and the notion of separating the communication and traffic monitoring responsibilities (as introduced in [26]). A *virtual trip line* (VTL) is a line segment in geographic space that, when crossed, triggers a client's location update to the traffic monitoring server. More specifically, it is defined by

$$[vtlid, x_1, y_1, x_2, y_2, d]$$

where $vtlid$ is the virtual trip line ID, x_1, y_1, x_2, and y_2 are the (x, y) coordinates of two line endpoints, and d is a default direction vector (e.g., N-S or E-W). The default direction vector encodes the valid direction in which the virtual trip line can be crossed. This directionally specific attribute can be used to reject location updates from vehicles crossing VTLs in the opposite direction, which can occur due to GPS errors and dense road networks. Also in case that a single VTL covers both directions on highways if it is long enough (to cover both northbound and southbound, or westbound and eastbound), the clients detect the direction from two successive coordinates and simply code the direction into 0 or 1 based on default direction vector.

When a vehicle traverses the trip line, its measurement update includes the time, trip line ID, speed, and the direction of crossing. The trip lines are pre-generated, downloaded, and stored in clients. To check any crossings, we set the sampling period of a single-chip GPS/A-GPS module in each smartphone and retrieve the position readings. Since our setup did not provide speed information, we calculate the mean speed using two successive location readings (in our implementation, every 3 seconds). The client software registers the task for checking the traversal of trip lines as an event handler for GPS module location updates, which is automatically invoked whenever a new position reading becomes available. As an example of required storage and bandwidth consumption, consider the San Francisco Bay Area, the total road network of which contains about 20,000 road segments, according to the Digital Line Graph 1:24K scale maps of the San Francisco Bay Area Regional Database managed by USGS. Assuming that the system on average places one trip line per segment this results in 166KB of storage.

Virtual trip lines control disclosure of location updates by sampling in space rather than sampling in time, since clients generate updates at predefined geographic locations (compared to sending updates at periodic time intervals). The rationale for this approach is that at specific locations, traffic information is more valuable and certain locations are more privacy-sensitive than others. Through careful placement of trip lines, the system can thus better manage data quality and privacy than through a uniform sampling interval. In addition, the ability to store trip lines on the clients can reduce the dependency on trustworthy infrastructure for coordination.

A. Strengths

The VTL concept can be extended to provide several additional benefits. First, as will be discussed throughout the article, it allows system designers to choose several different options for privacy protection. The levels of privacy protection range from forcing the location sampling in sensitive areas to achieving guaranteed privacy via k-anonymous cloaking. Second, for a given number of location updates from drivers, the VTL paradigm allows system designers to predefine measurement locations for high-value updates. For example, location updates from low priority residential streets can be avoided. Third, the use of VTLs removes the need for map matching the measurement update to road segments, since each VTL is already associated with a road segment. Fourth, system designers can embed traffic alerts or warnings on VTLs by piggybacking on the system's acknowledgement packet which responds to a user's location update. For example, VTLs may be defined with location descriptors associated with school zones, construction zones, or icy roads. Fifth, we can define a timer attribute for each VTL which specifies the allowable latency for each measurement. Thus, increasing the timer on a VTL allows users to delay the measurement report time, which aids in the prevention of adversarial tracking. Sixth, we can dynamically turn on/off VTLs depending on the time of day and congestion levels. Also around construction sites or detours, one can dynamically place more VTLs.

B. Virtual Trip Line Measurements

Noisy GPS readings can be filtered either on the client side or the server side. Server side processing can allow for a computationally expensive algorithm to filter out noisy GPS readings, for example using map-matching algorithms. However, it requires clients to send detailed traces to a server, which incurs increased network bandwidth consumption and privacy concerns. Instead we address filtering on the client, with the specific goals of subsampling GPS readings to reduce the frequency of trip line measurement computations (i.e., checking whether the line between two GPS readings intersects with any trip lines), and removing the need of any client side or server side map-matching algorithm, which is a computationally expensive algorithm for resource constrained devices.

We have observed that GPS position error can create false VTL crossings and inaccurate VTL velocity measurements in the following cases:

- *GPS position error.* When a vehicle stops near a trip line, error in the GPS position can create successive position measurements with a zigzag pattern over the VTL, which can lead to multiple false trip line crossings. These crossings can be eliminated by requiring a minimum distance between successive GPS readings.
- *Intermittent GPS.* When the time interval between two GPS positions becomes large (e.g., due to lost GPS signal), the inferred trajectory connecting these two location measurements no longer describes the actual movement of a vehicle. To eliminate false trip line crossings by this type of unrealistic trajectory, an upper bound of time gap between successive GPS samples is required.
- *Infeasible speed.* In areas prone to high GPS position error (e.g., urban areas with high-rise buildings), the speed computed from a finite difference approximation of the successive positions (required by the GPS receiver in our implementation) becomes infeasible. We refer to these errors as *speed glitches* in the remainder of the article.

Algorithm 1 below describes in detail our implementation of a light-weight client filtering algorithm to treat the common situations above. The algorithm proceeds as follows. First, if the GPS sample l is the first update, it is simply saved to *CurrLocationFiltered* (line 4-7). Without a previous update, we cannot compute the speed or heading of the current update or confirm it as valid. Assuming a previous update exists, the validity of the next update can be determined based on the computed speed and the temporal/spatial gap from previously filtered GPS reading called *PrevLocationFiltered*. We consider the current location invalid if it is updated long after the previous update (line 10-14), if it has not traveled a minimum distance (line 16-18, e.g., stopped at the traffic signal), or if has a speed glitch (line 19-30).

Additionally, we maintain two reference points, *LastGoodRefPoint* and *LastBadRefPoint*. If a series of locations have speed glitches against *LastGoodRefPoint*, but do not have speed glitches against *LastBadRefPoint*, we consider *LastBadRefPoint* and the most recent location in the series as valid (line 22-26). Next, the location update after the validity check is injected to a smoothing filter (called *SmoothingFilter* in algorithm 1), which is implemented by an exponentially-weighted moving average lowpass filter (line 24, 31). The smoothing filter produces a smoothed version of speed profile by cutting off abrupt speed changes. The final step is used to reduce the computational overhead created by the frequent checking of virtual trip line crossings on the output of algorithm 1. Instead of returning a location update at the maximal rate allowed by the GPS receiver, we return a location update only after every T seconds, which is encoded by the function *checkReportingInterval* (line 25, 33). A larger T makes computation of trip line crossings more efficient, but if it becomes too large, valid trip line crossings can be missed and false trip line crossings can be computed.

Algorithm 1 Tripline Crossing Detection Algorithm

```
1:  $\theta$ = thresholdToSwitchBadToGood
2:  $T$ = subsampling interval
3:  for all GPS sample $l$ do
4:      if PrevLocationFiltered is null then
5:          CurrLocationFiltered = LastGoodRefPoint = $l$;
6:          LastLocationUpdateTimestamp = $l.t$; goto TripLineChecking;
7:      end if
8:      TimeGap = $l.t$ - LastLocationUpdateTimestamp;
9:      LastLocationUpdateTimestamp = $l.t$;
10:     if TimeGap is too large then
11:         LastGoodRefPoint = $l$; LastBadRefPoint = null; $n$ = 0;
12:         CurrLocationFiltered = $l$; PrevLocationFiltered = null;
13:         goto TripLineChecking;
14:     end if
15:     Calculate speed against LastGoodRefPoint;
16:     if a vehicle has not moved far enough then
17:         LastBadRefPoint = null; $n$ = 0; CurrLocationFiltered = null;
18:         goto TripLineChecking;
19:     else if speed glitch is $true$ then
20:         Re-calculate speed against LastBadRefPoint;
21:         if speed glitch is $false$ then
22:             if ++$n$ is greater than $\theta$ then
23:                 $n$ = 0; LastBadRefPoint = null; LastGoodRefPoint = $l$;
24:                 filteredLoc = SmoothingFilter(LastBadRefPoint, $l$);
25:                 CurrLocationFiltered = checkReportingInterval(filteredLoc, $T$);
26:             end if
27:             goto TripLineChecking;
28:         end if
29:         LastBadRefPoint = $l$; goto TripLineChecking;
30:     end if
31:     $n$ = 0; filteredLoc = SmoothingFilter(LastGoodRefPoint, $l$);
32:     LastBadRefPoint = null; LastGoodRefPoint = $l$;
33:     CurrLocationFiltered = checkReportingInterval(filteredLoc, $T$);
34:     // TripLineChecking
35:     if both CurrLocationFiltered and PrevLocationFiltered not null then
36:         traj = SetTrajectory(PrevLocationFiltered, CurrLocationFiltered);
37:         for all tripline $j$ in each tile($i$) do
38:             if tile($i$).status is valid then
39:                 triplineCrossed = CheckCrossing(tripline $j$, traj);
40:                 if triplineCrossed is $true$ then
41:                     Compute speed and heading with traj for triplineMeasurement;
42:                 end if
43:             end if
44:         end for
45:     end if
46:     if CurrLocationFiltered is not null then
47:         PrevLocationFiltered = CurrLocationFiltered;
48:     end if
49: end for
```

The output returned from the algorithm 1 is then used by the software routine that computes virtual trip line crossings from consecutive filtered GPS positions (line 34-48). We check if any line defined by two end positions of each trip line intersects with a trajectory (built by two consecutive filtered GPS positions) in a two dimensional space. If crossed, the algorithm returns a tripline measurement including trip line ID, speed, heading, and timestamp information. All trip lines in downloaded tiles are tested, but limited to valid trip lines. Validity of trip lines can be subject to a combination of trip line's expiration time and user's privacy guidelines.

Discussion. The most challenging situation potentially experienced by the above algorithm occurs when the sampling frequency is too slow given the road geometry, and the route driven. The following example of a missed virtual trip line at an intersection illustrates this challenge. During a right turn maneuver at an intersection, if the position is sampled infrequently, then the two consecutive location updates may occur on two different roads, with one update occurring in the middle of the road before the right turn, and one update occurring in the middle of the road after the right turn. Then the straight line segment connecting these two points does not follow the road geometry, and any virtual trip lines placed near the intersection on either road segment will be missed. Moreover, if the sampling interval becomes large enough, the line segment between two consecutive location updates may intersect with virtual trip lines on the road segments not driven by the reporting vehicle, generating false measurements. This problem arises because of our removal of the computationally intensive map matching algorithm.

IV. Architecture Designs

We present two different architectures, one focused more on traffic estimation accuracy with probabilistic privacy preservation and an improved version that achieves guaranteed privacy using a k-anonymous temporal cloaking. The main purpose of temporal cloaking is to prevent an adversary from compromising anonymity, even in a very low user participation scenario. First, section IV-A describes the common parts for both architectures, then particular changes for each architecture follow in section IV-B and IV-C.

A. Achieving Authenticated but Anonymous Data Collection

In order to achieve the anonymization of measurement uploads from clients while authenticating the sender of the measurements, we split the actions of authentication and data processing into two different entities, which we call the ID proxy server and the traffic monitoring server. By separately encrypting the identification information and the sensing measurements (i.e., trip line ID, speed, and direction) with different keys, we prevent each entity from observing both the identification and the sensing measurements.

Figure 1 shows the resulting system architecture. It includes four key entities: probe vehicles with the cell phone handsets, an ID proxy server, a traffic monitoring service provider, and a VTL generator. Each probe vehicle carries a GPS-enabled mobile handset that executes the client application. This application is responsible for the following functions: downloading and caching trip lines from the VTL server, detecting trip line traversal, and sending measurements to the service provider. To determine trip line traversals, probe vehicles check if the line between the current GPS position and the previous GPS position intersects with any of the trip lines in its cache. Upon traversal, handsets create a VTL measurement including trip line ID, speed readings, timestamps, and the direction of traversal and encrypt it with the VTL server's public key. Handsets then transmit this measurement to the ID proxy server over an encrypted and authenticated communication link set up for each handset separately. The handset and the ID proxy server share an authentication key in advance.

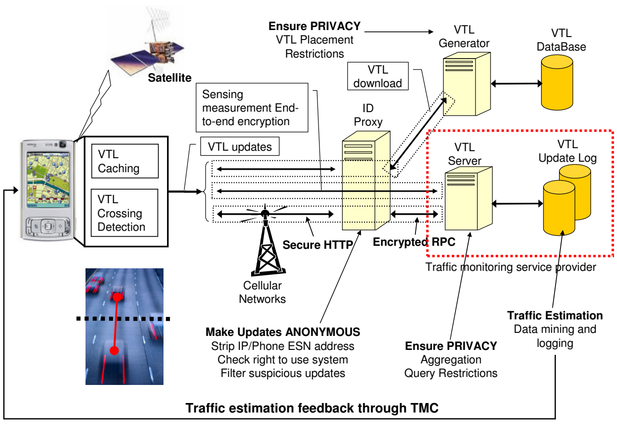


Fig. 1. Virtual Trip Line: Privacy-Preserving Traffic monitoring System Architecture.

The ID proxy server's responsibility is to first authenticate each client to prevent unauthorized measurements and then forward anonymized measurements to the VTL server. Since the VTL measurement is encrypted with the VTL server's key, the ID proxy server cannot access the VTL measurement content. It has knowledge of which phone transmitted a VTL measurement, but no knowledge of the phone's position. The ID proxy server strips off the identifying information and forwards the anonymous VTL measurement to the VTL server over another secure communication link.

The VTL server aggregates measurements from a large number of probe vehicles and uses them for estimating traffic conditions. The VTL generator determines the position of trip lines, stores them in a database, and distributes trip lines to probe vehicles when any download request from probe vehicles is received. Similar to the ID proxy server, each handset and the VTL generator share an authentication key in advance. The VTL generator first authenticates each download requester to prevent unauthorized requests and can encrypt trip lines with a key agreed upon between the requester and the VTL generator.[1] Both the download request message and the response message are integrity protected by a message authentication code.

Discussion. The above architecture improves location privacy of probe vehicle drivers through several mechanisms. First, the VTL server must follow specific restrictions on trip line placements that we will describe in section V-B. This means that a handset will only generate measurements in areas that are deemed less sensitive and not send any information in other areas. By splitting identity-related and location-related processing, a breach at any single entity would not reveal the precise position of an identified individual. A breach at the ID proxy would only reveal which phones are generating measurements (or are moving) but not their precise positions.

[1] While VTL positions are not highly sensitive, encryption reduces the possibility of timing analysis (see section VII-A).

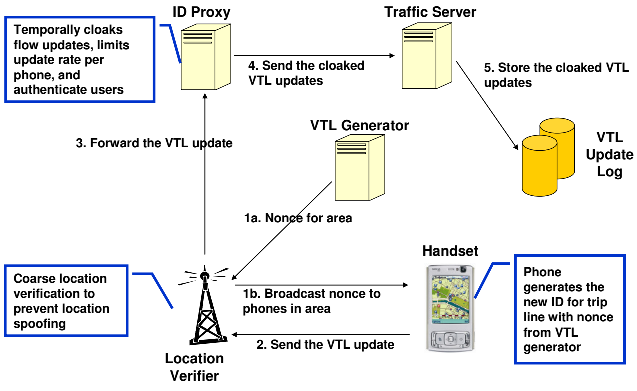

ID Proxy
Traffic Server
Temporally cloaks flow updates, limits update rate per phone, and authenticate users
4. Send the cloaked VTL updates
5. Store the cloaked VTL updates
VTL Update Log
3. Forward the VTL update
VTL Generator
1a. Nonce for area
Handset
Coarse location verification to prevent location spoofing
1b. Broadcast nonce to phones in area
2. Send the VTL update
Phone generates the new ID for trip line with nonce from VTL generator
Location Verifier

Fig. 2. Distributed Architecture for VTL-based Temporal Cloaking.

Similarly, a breach at the VTL server would provide precise position samples but not the individual's identities. Separating the VTL server from the VTL generator prevents active attacks that modify trip line placement to obtain more sensitive data. This is, however, only a probabilistic guarantee because tracking and eventual identification of outlier trips may still be possible. For example, tracking would be straightforward for a single probe vehicle driving along on empty roadway at night. The outlier problem in sparse traffic situations can be alleviated by changing trip lines based on traffic density heuristics. Trip lines could be locally deactivated by the client based on time of day or the clients speed. They could also be deactivated by the VTL generator based on traffic observations from other sources such as loop detectors. At the cost of increased complexity, the system can also offer k-anonymity guarantees regardless of traffic density. We will describe this approach next.

B. Guaranteeing K-Anonymity at Low Density Using Temporal Cloaking

We now demonstrate how virtual trip lines can help computing k-anonymous VTL measurements via temporal cloaking without using a single trusted server. Motivated by a well-known concept called a secret splitting scheme, we distribute secret information through multiple parties so that no central entity has complete knowledge of all three types of information: location, timestamp, and identity information. In doing so, we focus on minimizing any possible degradation of traffic information quality introduced by the information splitting scheme.

We propose a distributed VTL-based temporal cloaking scheme that reduces timestamp accuracy to guarantee a degree of k-anonymity in the dataset accumulated at the VTL server. This provides a stronger privacy guarantee than probabilistic privacy, since it prevents the tracking or reidentification of an individual phone even when user participation is very low. The key challenge in applying temporal cloaking is to conceal the locations of the probe vehicles from the cloaking entity. To calculate the time interval for probe vehicles at the same location, the cloaking entity typically needs access to the detailed records of each data subject [20], [40], which itself can raise privacy concerns.

Entity	Role	ID	Location	Time
Handset	Sense	Yes	Accurate	Accurate
Verifier	Broadcast VTL ID updates	Yes	N/A	Accurate
ID Proxy	Anonymize and cloak	Yes	N/A	Accurate
Traffic Server	Compute traffic	No	Accurate	Cloaked

TABLE I

SPLITTING OF ROLES AND SENSITIVE INFORMATION ACROSS ENTITIES.

Using virtual trip lines, however, it is possible to execute the cloaking function without access to precise location information. The cloaking entity can aggregate measurements by trip line ID, without knowing the mapping of trip line IDs to locations. It renders each measurement k-anonymous by replacing the measurement timestamp with a time window during which at least k measurements were generated from the same VTL (i.e., $k-1$ other phones passed the VTL). In effect, k VTL measurements are aggregated into a new measurement $(vtlid, \frac{s_1 \ldots s_k}{k}, \max(t_1 \ldots t_k))$, where s_i denotes the speed reading of each VTL measurement i. Since now k-phones generate the same measurement, it becomes harder to track one individual phone. The cloaking function can be executed at the ID proxy server, if handsets add a VTL ID to the measurement that can be accessed by the ID proxy server.

Beyond the cloaking function at the ID proxy server, two further changes are needed in the architecture to prevent an adversary from obtaining the mapping of VTL IDs to actual VTL locations. The system uses two techniques to reduce privacy leakage in the event of phone database compromises. First, the road network is divided into tiles, and phones can only obtain the trip line ID to location mapping for the area in which the phone is located. This assumes that the approximate position of a phone can be verified (for example, through the cellular network). Second, the VTL server periodically randomizes the VTL ID for each trip line and updates phone databases with the new VTL IDs for their respective location.

This leads to the extended distributed architecture depicted in Figure 2, in which again no central entity has knowledge of all three types of information: location, timestamp, and identity information. As before, VTL measurements from phones to the ID proxy server are encrypted, so that network eavesdroppers do not learn position information. It first checks the authenticity of the message and limits the upload rate per phone to prevent spoofing of measurements. It then strips off the identification information and forwards the anonymous measurement to the traffic server. With knowledge of the mapping of VTL IDs to locations, the traffic server can calculate road segment travel times. In this architecture, the ID proxy server cloaks anonymous measurements with the same VTL ID before forwarding to the traffic server. It also requires a location verification entity, which can coarsely verify phone location claims (e.g., in range of a cellular base station) and distribute the VTL ID updates to only the phones that are actually present within a specified tile. Table I summarizes the roles of each entity and how information is split across them.

The temporal cloaking approach can be vulnerable to spoofing attacks unless it is equipped with proper protection mechanisms. For instance, malicious clients can send a large number of measurements to shorten the cloaking time window. To prevent this denial of service attack, the ID proxy server limits the upload rate per phone.

To reduce network bandwidth consumption of the periodic VTL updates, clients can independently update the VTL IDs based on a single nonce per geographic area (tile). The VTL generator generates the nonces using a cryptographically secure pseudo random number generator and distributes each nonce and its expiration time to the clients currently in the tile area. Both clients and server can then compute $VTLID_{new} = h(nonce, VTLID_{old})$, where h is a secure hash function such as SHA. Then clients update the ID and the expiration time of each VTL in the current tile. In case that clients do not know the old ID (for example, as they have missed some updates or are new to a tile), the VTL generator still allows clients to download the set of whole VTLs with their new IDs in the tile. Each VTL has an expiration time beyond which its ID becomes invalid. If the connection is accidentally lost during downloading VTLs or the nonce, clients retry n times more until a successful downloading. The incomplete downloading can be easily checked by the header that includes the total number of VTLs in the corresponding tile (in our implementation). The expiration time of each VTL is used to synchronize the traffic server and clients. Clients decide whether or not to apply the ID update (using the nonce currently downloaded from the VTL generator), depending on whether the current ID of VTLs expires or not. Thus the synchronization based on the expiration time prevents clients from reapplying the ID update to VTLs that are already updated, so that it helps the procedure for calculating $VTLID_{new}$ idempotent.

Temporal cloaking fits well with the travel time estimation method used in the VTL system because the mean speed calculation does not depend on accurate timestamp information. To estimate the travel time, the traffic server calculates the mean speed for a trip line only based on the speed information in the VTL measurements. Typically, the travel time would be periodically recomputed. The use of temporal cloaking adaptively changes this mean speed calculation interval so that at least k phones have crossed the trip line. If k is chosen large, it reduces the update frequency. The rationale for temporal cloaking is that real-time traffic incident information such as congestion, potholes, and accidents requires more accurate location accuracy than timestamp accuracy. Since temporal information can be relaxed to provide enhanced user privacy as long as the monitoring events change relatively slowly, temporal cloaking can be generally applicable to other kinds of incident reports.

C. Balancing Privacy and Accuracy Requirements

The temporal cloaking architecture has several drawbacks in terms of real-time traffic estimation. First, since the ID proxy server needs to wait until it receives k VTL measurements, the system may fail to reflect brief events and incur unavoidable delay. This impact increases when a larger k is chosen. Second, in order to offer k-anonymity guarantees regardless of user participation rates, the system complexity is increased. Third, when the k measurements are averaged over a large period of time, the resulting measurement cannot be directly integrated into traffic estimation algorithms relying on the dynamics of traffic flow, which are commonly used in the transportation engineering community. To overcome these limitations, we propose an alternative architecture which focuses on real time traffic monitoring accuracy by relaxing the privacy requirements down to probabilistic privacy guarantee. The main idea is to remove k-anonymous temporal cloaking to allow the traffic server to receive k individual anonymous VTL measurements. Thus, at the cost of sacrificing a privacy guarantee, we alleviate system complexity, and enable the use of flow based traffic estimation algorithms, described next.

Traffic Estimation Algorithm. We briefly outline the velocity estimation algorithm developed in [42] and implemented in our accuracy–centric architecture. The estimation algorithm combines VTL measurements with a traffic flow model to produce an estimate of the average velocity field along the roadway. The flow model is based on the seminal *Lighthill–Whitham–Richards* (LWR) [32], [37] *partial differential equation* (PDE), which is given by:

$$\frac{\partial \rho(x,t)}{\partial x} + \frac{\partial Q(\rho(x,t))}{\partial x} = 0 \qquad (1)$$

where $\rho(x,t)$ is the vehicle density (so that $\int_a^b \rho(x,t)dx$ expresses the total number of vehicles on the roadway between a and b), and $Q(\cdot)$ is the vehicle flux as a function of the density. By exploiting a relationship between the density of vehicles and their average velocity, this model can be transformed into a discrete, nonlinear, nondifferentiable velocity evolution equation. In state space form the model becomes:

$$v^n = \mathcal{M}\left(v^{n-1}, \theta^{n-1}\right) + \eta^n \qquad (2)$$

where $v^n \in \mathbb{R}^m$ is the vector of average velocity at time n on each of the m discrete road segments in the transportation network, $\mathcal{M}(\cdot,\cdot)$ is the discrete velocity evolution equation with model parameters θ^n, and η^n represents the process noise.

The observation model is as follows:

$$y^n = \mathbf{H}^n v^n + \chi^n \qquad (3)$$

where y^n is the vector of VTL measurements at time n, $\mathbf{H}^n$ is a linear observation operator which maps the location of the virtual trip line measurements to the corresponding elements in the velocity vector v^n, and χ^n is the error introduced due to sampling errors and GPS errors.

The estimation problem is then solved using an extension of Kalman filtering known as *ensemble Kalman filtering* [13], to overcome the nonlinearity and nondifferentiability of $\mathcal{M}(\cdot,\cdot)$.

V. Experimental Evaluation

We have fully implemented the probabilistic privacy architecture with *ensemble Kalman filtering* (in section IV-A and IV-C) and used this implementation for a one day field experiment called *Mobile Century*. The data collected during the experiment were used to reevaluate our k-anonymous temporal cloaking and its privacy-relaxed version for accuracy improvement. A detailed description regarding the system implementation can be found in the earlier work [25].

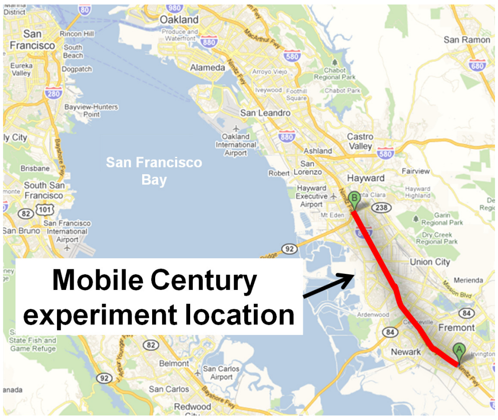


Fig. 3. Test Road Segment in Mobile Century.

A. The Mobile Century Experiment

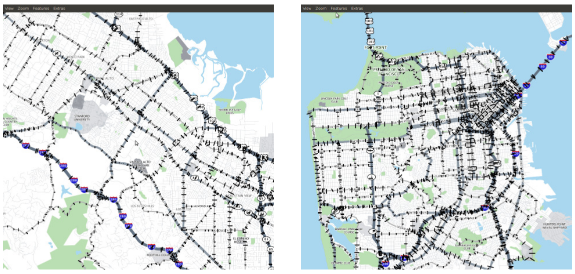
A large-scale experiment was conducted to demonstrate the feasibility of cell phone-based travel time estimation in practice. The event, named *Mobile Century* was a one-day field experiment which included 100 vehicles continuously driving a stretch of freeway in northern California. A complete description of the experiment and an analysis of the data collected during the experiment is described in [22]. We summarize the important features of the experiment here, and the data is available for download at [1].

The vehicles were equipped with cell phones running a mobile client which allowed virtual trip line measurements to be collected from the devices. Additionally, log files stored locally on the phones recorded the position and speed of the vehicles for analysis after the experiment. The 100 vehicles were split into three groups, each group drove overlapping segments of an 11 mile stretch of I880 near Hayward, California as shown in figure 3, but each group used separate entrance and exit ramps. The stretch of roadway was selected because of a recurring bottleneck which causes severe afternoon congestion in the northbound direction. In order to capture the travel times of vehicles not participating in the experiment, high definition video cameras were set up on several overpasses to record northbound traffic. From this video data, we used license plate reidentification to measure the travel times across a 6.5 mile subsection of the experiment. The travel times ranged between seven and 20 minutes during the day.

After the experiment concluded, it was identified that 77 of the cell phones running the experimental software were able to properly record the probe vehicles' positions and velocities, which generated 2200 vehicle trajectories across the experiment site during the eight hour experiment. These trajectories make up between 0% and 5% of the total traffic flow depending on the time of day [22]. Using the data obtained from these vehicles, we were able to assess the impact of virtual trip line spacing and the number of participating vehicles on the accuracy of computing travel times.

B. Trip Line Placement

We use the combination of the following techniques to determine the positions of VTLs.



(a) Palo Alto (b) San Francisco

Fig. 4. Example of Virtual Trip Lines Placements.

Exclusion Area via Road Category. Privacy can be significantly improved by restricting trip line placement to high traffic roadways, such as highways and arterials, which are also typically less sensitive areas. We extend the concept of exclusion area in our earlier work by restricting placement to these roadways. To determine our placement, we use the road category information provided by the Navteq street database, which classifies each road segment from 1 (highest capacity roads) to 5 (the lowest capacity roads). We only place VTLs on road categories 1 to 3, which avoids trip line placement in residential areas. Figures 4(a) and 4(b) show examples of virtual trip lines placements in Palo Alto and San Francisco respectively. This approach prevents an adversary from identifying the precise origin and destination of the tracked user in many situations, but it cannot deliver guaranteed privacy protection when sensitive locations are on high capacity road segments.

Equidistant Spacing with Data Obfuscation. This approach takes as input a network graph of road segments in the category of our interest as explained above. For each road segment, defined by stretches of roadway between intersections or merges/diverges, the algorithm places equidistant trip lines orthogonally to the road. A large spacing makes it harder to track anonymous users as we demonstrated in our earlier study [25]. In the study, we focus the minimum spacing constraint on straight highway scenarios, in which more regular traffic flows increase the tracking risks. Minimum spacing for longer road segments is determined based on a tracking uncertainty threshold. Recall that to prevent linking compromises, an adversary should not be able to determine with high confidence that two anonymous VTL measurements were generated by the same handset. Tracking uncertainty defines the level of confusion that an adversary encounters when associating two successive anonymous VTL measurements to each other. We define tracking uncertainty as the entropy $H = -\sum p_i \log p_i$, where p_i denotes the probability (from the adversary's perspective) that anonymous VTL measurement i at the next trip line was generated by the same phone as a given anonymous VTL measurement at a previous trip line. The probability p_i is calculated based on an empirically derived pdf model that takes into account the time difference between the predicted arrival time at the next trip line and the actual timestamp of VTL measurement i. We fit an empirical pdf of time deviation with an exponential function, $\hat{p}_i = \frac{1}{\alpha} e^{-\frac{t_i}{\beta}}$, where we obtain the values of α and β by using unconstrained

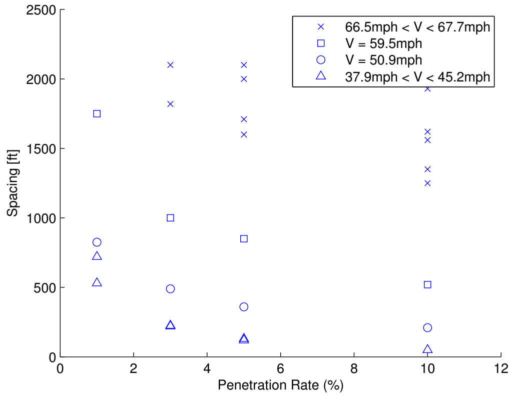


Fig. 5. Minimum Spacing Constraints for Straight Highway Section.

nonlinear minimization. Higher penetration rates lead to more VTL measurements around the projected arrival time, which decreases certainty. As spacing increases, the likelihood that speeds and the order of vehicles remain unchanged decreases, leading to more uncertainty.

We empirically validate these observations through simulations using the PARAMICS vehicle traffic simulator [3]. Figure 5 depicts the minimum spacing required to achieve a minimum mean tracking uncertainty of 0.2 for different penetration rates and different levels of congestion (or mean speed of traffic). We choose a reasonably low uncertainty threshold, which ensures to an adversary a longer tracking that could have privacy events such as two different places (e.g., origin and destination). Two recent studies [31], [26] observe about 15 minutes as a median trip time. The uncertainty value of 0.2 corresponds to an obvious tracking case in which the most likely hypothesis has a likelihood of 0.97. The penetration rates used were 1%, 3%, 5% and 10%. To evaluate different levels of congestion, we used traces from seven 15 min time periods distributed over one day. We also used three different highway sections (between the junction of CA92 and the junction of Tennyson Rd., between the junction of Tennyson Rd. and the junction of Industrial Rd., and between the junction of Industrial Rd. and the junction of Alvarado-Niles Rd.) to reduce location-dependent effects. The simulations show that the needed minimum spacing decreases with slower average speed and higher penetration rate. The clear dependency of the tracking uncertainty on the penetration rate and the average speed allows creating a model that provides the required minimum spacing for a given penetration rate and the average speed of the target road segment.

VI. Results

This section first evaluates the performance of the tripline crossing detection algorithm presented in section III. Then, we analyze the travel time estimation accuracy and privacy preservation of our spatial sampling approaches using virtual trip lines. Spatial sampling approaches to be evaluated here include *k*-anonymous temporal cloaking and its privacy-relaxed version where we strip off the requirement of guaranteed privacy via temporal cloaking. The former is the proposed scheme but the latter is still meaningful in that it is a baseline technique to be compared with the proposed scheme and a less complex system with an acceptable privacy protection in the real world.

# Sample Types	Number of Samples
Intermittent GPS samples	5
Zigzag GPS samples	745
Speed glitches	13
Good GPS samples	894

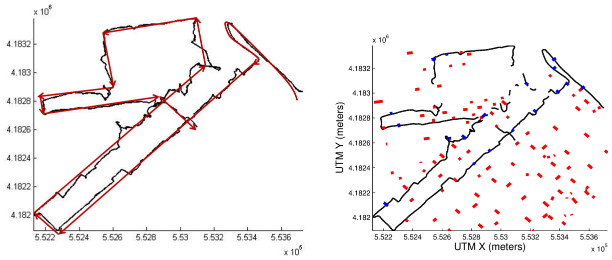
TABLE II
Removals of GPS sample errors help reduce the frequency of tripline crossing checks.

	Detection	False Alarm
San Francisco	47%	11%
San Jose	98%	3.6%

TABLE III
The worst GPS accuracy in SF approximately degrades the performance of tripline crossing detection algorithm by half.

A. *VTL Measurement Accuracy*

Tripline Crossing Detection. We observed that GPS position error creates false VTL crossings and drops the detection performance in experimental GPS traces collected in San Francisco downtown, as shown in figure 6(a). Collected traces cover Market st., Mission st., Pine st., Bush st., and Washington st., where the worst GPS positioning accuracy is expected, due to highrise buildings and cloudy weather. Figure 6(b) illustrates the filtered GPS trace, a smoothed version of original GPS trace after intermittent GPS samples, zigzag GPS samples, and speed glitches are removed by algorithm 1. Table II summarizes the number of removed samples corresponding to each case. The number of “Good GPS samples” are reported 894 samples in our algorithm, but its definition is based on whether the speed of two successive filtered locations lies within a valid range. Thus if a map-matching algorithm is additionally applied to the collected trace, the number of “Good GPS samples” should be larger. To observe the dependency of the presented algorithm on GPS positioning accuracy and wireless connectivity, we collected traces in San Jose downtown and measured the detection probability and false alarm probability of VTL crossings for both cities. In San Jose (where better GPS accuracy is expected than San Francisco), the presented algorithm detects 98% of VTLs placed on the route; only 3.6% of reported crossings were false. However, the detection probability was significantly degraded to 47% in San Francisco, and the false alarm probability increased up to 11%. Dense road network in San Francisco downtown makes the situation worse even with a few meter GPS error. In this experiment, there was a very slight change in the number of false alarms and detections for the two different situations (with the algorithm and without the algorithm) since removals of GPS samples do not coincide with the VTL locations in the collected traces. However, the benefit of these removals should be observed if more traces are tested. To see how many false crossings these removals would



(a) GPS Traces: ground truth (red) vs. unfiltered (black).

(b) VTLs and Filtered GPS Traces (black).

Fig. 6. In the right figure, red colored lines denote VTLs placed in downtown while blue lines denote detected VTL crossings.

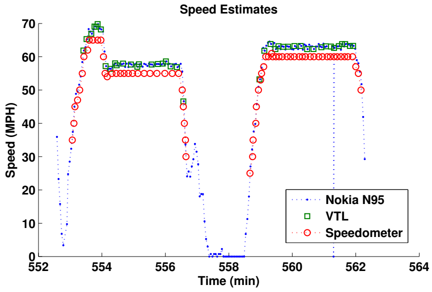


Fig. 7. Comparison of the speed measurements recorded from the N95 (dots), the VTLs (boxes) and the vehicle speedometer (circles) as a function of time.

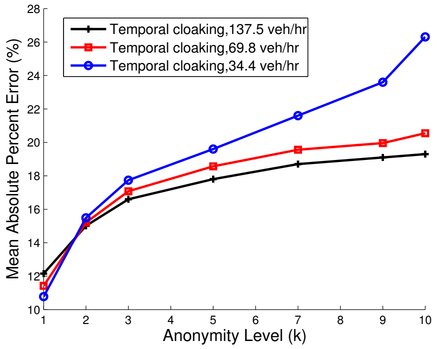


Fig. 8. Travel Time Accuracy versus Anonymity Level using 10 VTLs/mile.

potentially save in unfiltered GPS traces, we count the number of crossings that unfiltered and filtered GPS traces create on wrong road segments and compare with the counts. Unfiltered GPS traces create 17 crossings on wrong road segments while filtered GPS traces have 9 crossings. If we consider these wrong road segment crossings into false alarm probability assuming that VTLs are placed dense enough to coincide with these wrong crossings, unfiltered GPS traces have almost two times false alarm probability compared to filtered GPS traces. We find that the presented algorithm has two major benefits; it removes potential false alarms by removing GPS position errors and removing samples (almost 50% of GPS samples removed in this experiment) helps reduce the frequency of tripline crossing checking, thereby relieving the computation overhead.

GPS Speed Accuracy. Another field test was run to estimate the speed accuracy of a single cell phone carried onboard a vehicle. The experiment route consisted of a single 7 mile loop on I-80 near Berkeley, CA, and VTLs were placed evenly on the highway every 0.2 miles. Speed and position measurements were stored locally on the phone every 3 seconds, and speed measurements were sent over the wireless access provider's data network every time a VTL was crossed. The speed measurements were computed using two consecutive position measurements. In order to validate this calculation, the vehicle speed was also recorded directly from the speedometer on a laptop with a clock synchronized with the N95. In Figure 7, the speed measured directly from the vehicle speedometer is compared to the speeds measured by the VTLs and the speed stored in the phone log. Timestamp of each record denotes the elapsed time since midnight (of the experiment day). On average, the vehicle odometer reported a speed 3 mph slower than the GPS.

B. *Guaranteed Privacy via VTL-based Temporal Cloaking*

To evaluate the performance of VTL-based temporal cloaking, we compute its travel time estimation accuracy in offline mode with the collected traces from *Mobile Century*. The procedure for computing travel time consists of three steps. First, we divide the I-880 northbound highway segment (used in the experiment) into multiple sections, putting one VTL in the middle of each section. Second, we compute the speed profile for each section, where the speed profile denotes the change of mean speed over time. The mean speed is updated when the VTL on the section receives k-anonymous VTL measurements. Lastly, we compute the time taken to traverse each section and compute the sum from the first section to the last one. To compute the travel time for each section, we read the initial speed of the vehicle at the moment of entering the section from the speed profile of the section and let the vehicle follow the speed profile of the section until the vehicle exits the section.

To see the effect of k on travel time estimation accuracy, we vary k up to 10. In order to see the sensitivity of travel time estimation accuracy on penetration rates, we control the penetration rate by respectively using the full set of probe vehicles (about 137.5 veh/hr), half of them, and one fourth of them. Figure 8 shows that temporal cloaking achieves less than 18% travel time error using $k = 5$ and a probe rate of 137.5 veh/hr, which corresponds to about 2% penetration rate in the morning and about 1% in the afternoon [22]. The cases for $k = 1$ can be considered periodic sampling techniques with the same number of VTL measurements collected as temporal cloaking. Compared to a periodic sampling, the proposed scheme sacrifices about 5% accuracy to achieve the guaranteed privacy (k=5).

C. *Reconstructing VTLID-Location Mapping*

Concealing the mapping between each VTL ID and its location is a key enabler of temporal cloaking. However, the mapping can be partially reconstructed by an active attack at the level of the compromised ID proxy server. For example, let us consider a scenario in which a malicious ID proxy server performs an active attack with a small fraction of handsets and the VTL generator refreshes each VTL ID every 10

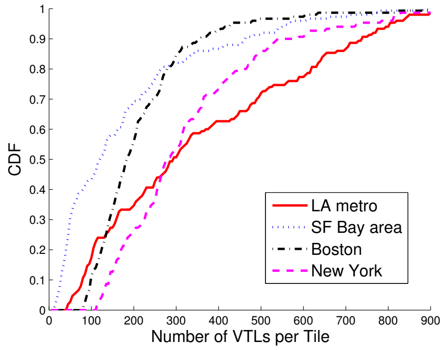


Fig. 9. The CDFs of number of virtual triplines per tile (8km by 8km) in different major cities in US.

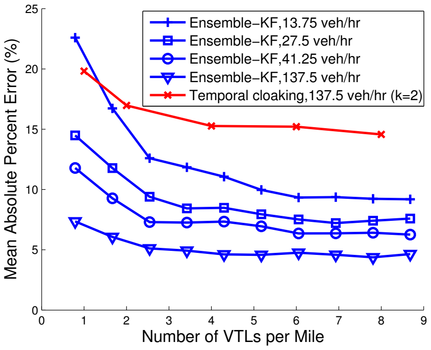


Fig. 10. Tradeoffs between number of virtual trip lines per mile and the travel time error for different values of the number of equipped vehicles sending measurements per hour.

minutes. Each compromised handset sends VTL measurements associated with random VTL identifiers and timestamps to the ID proxy server. Later, all VTL measurements can be cross-checked against GPS logs (containing GPS position with timestamp) collected separately by a compromised vehicle.

To evaluate the difficulty of reconstructing the VTL ID and location mapping that is randomly changed by a secure hash function and a nonce, we use the VTL database that contains all virtual trip lines placed over the United States. To build the database, we ran the automated algorithm explained in section V-B with an average spacing of 1000ft. The 90 percentile of tiles in SF Bay area have about 500 VTLs as shown in figure 9, so that the total length of roads covered by VTLs would be $500,000ft \simeq 94miles$. Following the attack described above, an ID proxy server would require about 14 vehicles (assumed to run 40mph) per tile to reveal the mapping of all VTLs. As more frequent VTL ID updates are used to randomize the mapping, the number of compromised vehicles required per tile should increase linearly for reconstruction. In Los Angeles metro and New York, for example, more number of compromised vehicles (equipped with handsets) are required to cover larger number of VTLs due to their more dense road networks.

D. Accuracy-Centric Architecture

In order to compute travel times from VTL data with temporal cloaking relaxed, we use a highway traffic estimation algorithm which estimates the average velocity field along the roadway as described in section IV-C. This algorithm ran live during the *Mobile Century* experiment, and has run live in northern California for now more than two years since it has been implemented in a traffic estimation engine at UC Berkeley. Travel times are then computed using this velocity field estimate, by considering the travel time a vehicle would experience by driving the velocity estimated by the algorithm.

For the numerical experiments presented next, a subset of virtual trip lines and a subset of the participating vehicles are selected for northbound I880 and the resulting measurements are fed into the velocity estimation algorithm. The impact of the virtual trip line spacing and the number of participating vehicles on the travel time accuracy are shown in Figure 10. Each curve corresponds to a different number of equipped vehicles, ranging from 13.75 veh/hr (10% of the 2200 trajectories) to 137.5 veh/hr (100% of the 2200 trajectories). Similarly, we adjusted the number of trip lines deployed on the experiment site, from nine trip lines to 99 trip lines in increments of 10. Figure 10 shows how improvements in accuracy can be achieved either by increasing the number of vehicles sending measurements or by increasing the number of locations where measurements are obtained from a fixed number of vehicles. In the case in which numerous vehicles are participating and the virtual trip line spacing is sparse, the experiment shows that it is possible to reconstruct travel times with less than 10% error while maintaining a high degree of anonymity for the participating users. Furthermore, the travel times can be computed without measurements of the travel times of the equipped vehicles, which would have required disclosure of the full vehicle trajectories.

Compared to temporal cloaking (at best when $k = 2$ and full probe vehicles used), the accuracy centric architecture enhances the travel time estimation error by almost 10% (achieving about 5% travel time estimation error when more than two VTLs are place per mile), which is again even better than periodic sampling techniques. For example, 2.5 VTLs per mile has about 2100ft spacing, which easily meets the minimum spacing constraint of 1750ft that maintains the tracking uncertainty less than 0.2 for roads where 1 to 10% penetration rates of probe vehicles run at the average speed of 0 to 60mph as shown in figure 5. The comparison between the accuracy centric architecture and temporal cloaking architecture demonstrates the price the system designer pays for privacy, which is about 10% accuracy reduction. Moreover, even when the penetration rate of probe vehicles for the accuracy centric architecture is 1/10 that of the temporal cloaking architecture, the accuracy centric architecture produces travel time estimates with lower error. So the guaranteed privacy comes at the cost of significantly more vehicles required to achieve the same level of accuracy.

In Figure 11, we show a comparison of the mean travel time obtained from our video data compared to the mean travel time computed using our spatial sampling approaches using virtual trip lines. This comparison corresponds to a mean absolute percent error of about 5% for the accuracy centric architecture, which was achieved using 100% of our equipped

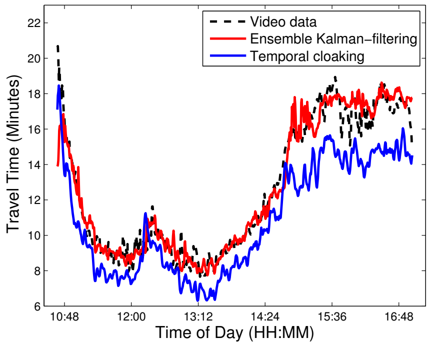


Fig. 11. Comparison between mean travel time obtained from video data, mean travel time obtained from temporal cloaking, and mean travel time obtained from the accuracy centric architecture.

vehicles and about 8.6 VTLs per mile. The largest error in this simulation occurs in the morning around 10:40 AM, with an error of about six minutes. The high travel times experienced by drivers at this time are caused by a five car accident. The estimation algorithm performs poorly here for two reasons. First, the traffic model used in the estimation algorithm does not predict accidents. Second, because the accident occurred at the beginning of the experiment, some of the equipped vehicles had not yet been deployed resulting in few measurements to correct the model. Throughout the rest of the day, the estimated travel times are significantly closer to the mean travel times. The Temporal cloaking approach used for the comparison achieves about 15% travel time error, where k is set to two and 100% of our equipped vehicles upload VTL measurements from 8 VTLs per mile.

VII. Discussion

We now discuss limitations and outlooks of our presented approaches as well as share lessons learned from the field operational deployment.

A. Security

The proposed architecture significantly improves privacy protection over earlier proposals, by distributing the traffic monitoring functions among multiple entities, none of which have access to location, timestamp, and identity records at the same time.

The system protects privacy against passive attacks under the assumption that only a single infrastructure component is compromised. One passive attack that remains an open problem for further study is timing analysis by the ID Proxy server or by network eavesdroppers between the Location verifier and the ID Proxy server. For the case of an adversary at the ID Proxy server, the adversary can hire multiple handsets (their IDs known to the adversary) and ask them to move around a target area. By comparing the GPS traces driven by those handsets with their trip line updates, the adversary can learn the exact trip line locations. In addition,the adversary could estimate the time needed to travel between any two trip lines from public travel time information on the road network. Then the adversary could attempt to match a sequence of observed VTL update message inter-arrival times to these trip line locations. This attack can be also conducted by network eavesdroppers passively observing the channel between the Location verifier and the ID Proxy server. One may expect that the natural variability of driving times provides some protection against this approach. Protection could be further strengthened against network eavesdroppers by inserting random message delays on the handset (client application) side. Under the temporal cloaking scheme, however, the ID proxy server also obtains trip line identifier information. If trip line identifier information is used for extended durations, an adversary may match them to actual VTL positions based on the sequence in which probe vehicles have passed them. This threat can be alleviated through frequent VTL ID updates. Quantifying these threats and choosing exact tile size and update frequency parameters to balance privacy and network overhead concerns remain open research problems.

The system also protects the privacy of most users against active attacks that compromise a single infrastructure component and a small fraction of handsets. It does not protect user privacy against injecting malware directly onto users' phones, which obtains GPS readings and transfers them to an external party. This challenge remains outside of the scope of this paper, because this vulnerability is present on all networked and programmable GPS devices even without the use of a traffic monitoring system. Instead, the objective of the presented architecture is to limit the effect of such compromises on other phones. For the temporal cloaking approach, compromised phones result in two concerns. First, an adversary at the ID proxy can learn the current temporary trip line IDs. To limit the effectiveness of this attack, the architecture periodically changes trip lines and verifies the approximate location of each phone so that a tile of trip line updates can only be sent to phones in the same location. Second, a handset could spoof trip line updates at a certain location to limit the effectiveness of temporal cloaking. Our proposed architecture already eliminates updates from unauthorized phones and can easily limits the update rate per phone and verify that the approximate phone position matches the claimed update. This renders extended tracking of individual difficult because it would either require a large number of compromised phones spread around the area in which the individual moves, or set of compromised phones that move together with the individual. The system could also incorporate other sanity checks and blacklist phones that deliver suspicious updates.

The same methods also offer protection against spoofing attacks that seek to reduce the accuracy of traffic monitoring data. The system does not offer full protection against any active attack on traffic monitoring accuracy, however. For example, a compromised ID proxy could drop messages to reduce accuracy. These challenges remain an open problem for further work.

As in any secret-splitting scheme, the proposed architecture cannot offer protection if adversaries within the different entities collude or if an adversary manages to break into multiple entities. Experience from current privacy violations has shown, however, that the vast majority are due to accidental disclosures or a single disgruntled or curious insider [19],

[33]. If implemented correctly, no individual insider would have access to more than one of the proposed entities, thus our secret-splitting architecture provides adequate protection against this important class of privacy violations.

B. Involvement of Cellular Networks Operators

While this work was based on cellular handsets, the question of how to improve location privacy within cellular networks themselves was outside of the scope of this work. The Phase II E911 requirements [2] mandate that cellular networks be able to locate subscriber phones within 150-300m 95% of the time, and A-GPS solutions should achieve similar accuracy. In addition, phones are identifiable through IMSI (International Mobile Subscriber Identity, in the GSM system) and operators typically know their owner's names and addresses. While precise phone location information is accessible, to our knowledge, it is not widely collected and stored by operators at this level of accuracy.

This work investigated how traffic monitoring services can be offered without access to sensitive location information. It was primarily motivated by third party organizations that currently do not yet have access to identity and location information and want to implement privacy-preserving traffic monitoring services. Our solution is general enough so that in actual implementations, different levels of involvement of network operators are possible. One case may be four separate organizations, each operating a different component of the system with no involvement of the network operator.[2] Another extreme case would be a cellular network operator creating separate entities within the company to protect itself against dishonest insiders and accidental data breaches of their customers records. Clearly, the first would be preferable from a privacy perspective, but in the end both lead to a significant improvement in privacy over a naive implementation.

C. Challenges in Arterial Roads Traffic Estimation

In comparison to highway traffic, arterials present additional challenges. The underlying flow physics that governs arterials is more complex (traffic lights often with unknown cycles, intersections, stop signs, parallel queues). While our work [10], [11], [42] explicitly derives techniques to reconstruct traffic from VTL type data, such a reconstruction becomes harder for arterials. Also, while macroscopic flow models such as the ones used in [10], [11], [42] exist and can be used for secondary networks [17], [39], their parameters are in general unknown or inaccessible and only documented for few cities, making their use impossible without going to the field and measuring them. In addition, even if they were known, the complexity of the underlying flows makes it challenging to perform estimation of the full macroscopic state of the system at low penetration rates. In light of these challenges, statistical approaches for characterizing a subset of the macroscopic state (for example travel times and aggregated speeds) have proved to work well and seem to be one of the only alternatives to traffic flow model based traffic reconstruction [15], [35]. Ongoing work has focused on sampling policies for arterial networks [23], [24].

D. The Mobile Millennium Field Operational Test

For one year starting in November 2008, a pilot project known as *Mobile Millennium* was deployed in Northern California. Residents of the Bay Area were able to download a traffic client on Java enabled mobile phones, which displayed real time traffic conditions while collecting virtual trip line data. Unlike the *Mobile Century* experiment described earlier, *Mobile Millennium* monitored traffic conditions on highways and arterials continuously throughout the year.

This pilot project highlighted a fundamental challenge for launching privacy preserving traffic monitoring systems using GPS data, which is to achieve high participation rates amongst the driving public when launching the system. Even with more than 5000 application downloads, only a few hundred users ran the *Mobile Millennium* application at any given time across a large geographic area. Thus, the resulting data from these devices was sparse both in space and time. At low participation rates, an architecture relying on temporal cloaking is insufficient to monitor real–time traffic conditions accurately due to the latency required for anonymity. At the same time, without temporal cloaking, reidentification of users at low participation rates becomes much easier. To overcome these difficulties at low data rates, *Mobile Millennium* was bootstrapped with additional data such as inductive loop detectors, fleet GPS data, radar data, and toll tag data to augment the traffic monitoring system. In turn, this launched additional research at UC Berkeley on traffic data fusion to assimilate traffic data from these data sources, which is still ongoing.

VIII. Related Work

Traffic monitoring applications based on a large number of probe vehicles have recently received much attention [28], however location privacy concerns have not been adequately addressed. The anonymization of sensing information has been the preferred solution in practical deployments [5], [6], [7]. Not surprisingly, recent analyses of GPS traces [27], [31], [41] have shown that naive anonymization by simply omitting identifiers from location dataset does not guarantee anonymity.

Stronger protection mechanisms have been investigated. K-anonymity [40] provides a guaranteed level of anonymity for a database, although some recent studies [30], [34] have identified weaknesses in k-anonymity. For location services, k-anonymity has led to the development of centralized architectures that temporally and spatially cloak location-based queries [16], [20]. Our work, in comparison, concentrates on providing privacy without requiring a single trustworthy entity.

There are many best effort approaches [9], [29] that degrade information in a controlled way before releasing it. These approaches can be implemented in a centralized or a decentralized architecture. Many best effort approaches successfully preserve the privacy of users in high density areas, but they

[2] The only limitation is that for temporal cloaking one of the identities needs to be able to approximately (at the level of a tile size) verify client location claims. This verification could be provided by a network operator but other forms of verification are also plausible.

do not guarantee privacy for low user density. Hoh et al. [27] propose an uncertainty-aware path cloaking algorithm that provides guaranteed privacy regardless of user density, but this again requires the existence of a trustworthy privacy server.

IX. Conclusions

This article described traffic monitoring system implemented on GPS smartphone platform. The system uses the concept of virtual trip lines to determine when phones reveal a location update to the traffic monitoring infrastructure. We demonstrated that the introduced scheme, Virtual Trip Lines, successfully addresses known weaknesses of probe vehicle based traffic monitoring. First, the VTL paradigm achieves strong anonymity through k-anonymous temporal cloaking. Virtual trip lines allow the application of temporal cloaking techniques to ensure k-anonymity properties of the stored dataset, without having access to the actual location records of phones. Second, they improve the accuracy of traffic monitoring. We show that the temporal cloaking leads to less than 5% reduction in the accuracy of travel time estimates for k values less than 7 compared to periodic sampling techniques and a privacy-relaxed version achieves 5% travel time estimation error using only 1-2% penetration rate. Third, VTLs enable a light-weight client algorithm for collecting VTL measurements, and we achieve the VTL crossing detection between 50% to 98% in downtowns while suppressing false alarm less than 11% without map-matching.

Acknowledgement

The authors would like to thank Kenneth Tracton, Carl Snellman, and Cynthia Kuo from Nokia Research Center for helping implement the system in the large scale experiments. The experiment was supported by the Caltrans. This study was supported in part by the US NSF under grant CNS-0524475.

References

[1] http://traffic.berkeley.edu/.

[2] http://www.fcc.gov/Bureaus/Wireless/.

[3] http://www.paramics-online.com.

[4] http://www.privacyrights.org/ar/ChronDataBreaches.htm.

[5] TeleNav. http://www.telenav.net/, 2004.

[6] Inrix. http://www.inrix.com/, 2006.

[7] Google map. http://www.google.com/mobile/maps/.

[8] Waze. http://www.waze.com/, 2009.

[9] A. Beresford and F. Stajano. Mix zones: User privacy in location-aware services. In *IEEE PerSec*, 2004.

[10] C. Claudel and A. A. Bayen. Lax-hopf based incorporation of internal boundary conditions into hamilton-jacobi equation. part II: Computational methods. in press, *IEEE Transactions on Automatic Control*, 2009.

[11] C. Claudel and A. Bayen. Lax-hopf based incorporation of internal boundary conditions into hamilton-jacobi equation. part I: Theory. in press, *IEEE Transactions on Automatic Control* 2009.

[12] X. Dai, M. Ferman, and R. Roesser. A simulation evaluation of a real-time traffic information system using probe vehicles. In *IEEE ITS*, pages 475–480, 2003.

[13] G. Evensen. *Data Assimilation: The Ensemble Kalman Filter*. Springer-Verlag, Berlin Heidelberg, 2007.

[14] M. Ferman, D. Blumenfeld, and X. Dai. A simple analytical model of a probe-based traffic information system. In *IEEE ITS*, pages 263–268, 2003.

[15] C. Furtlehner, J. M. Lasgouttes, and A. D. L. Fortelle. A belief propagation approach to traffic prediction using probe vehicles. In *IEEE ITS*, pages 1022–1027, 2007.

[16] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *IEEE ICDCS*, pages 620–629, 2005.

[17] N. Geroliminis and C. Daganzo. Macroscopic modeling of traffic in cities. *86th Annual Meeting TRB*, 2007.

[18] P. Gilbert, L. P. Cox, J. Jung, and D. Wetherall. Toward trustworthy mobile sensing. In *ACM HotMobile*, pages 31–36, 2010.

[19] L. Greenemeier, E. Malykhina, P. McGougall, A. Ricadela, and M. K. McGee. The high cost of data loss. http://www.informationweek.com/shared/printableArticle.jhtml?articleID=183700367, Mar 2006.

[20] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *ACM MobiSys*, 2003.

[21] M. Gruteser and B. Hoh. On the anonymity of periodic location samples. In *Proceedings of the Second International Conference on Security in Pervasive Computing*, 2005.

[22] J.-C. Herrera, D. Work, R. Herring, J. Ban, Q. Jacobson, and A. Bayen. Evaluation of traffic data obtained via GPS-enabled mobile phones: the Mobile Century experiment. *Transportation Research Part C*, 2009.

[23] R. Herring, A. Hofleitner, S. Amin, T. A. Nasr, A. A. Khalek, P. Abbeel, and A. Bayen. Using mobile phones to forecast arterial traffic through statistical learning. In *89th Annual Meeting TRB*, 2010.

[24] A. Hofleitner, R. Herring, T. Hunter, P. Abbeel, and A. Bayen. A learning and estimation approach towards arterial traffic monitoring. *In review.*, 2010.

[25] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J.-C. Herrera, A. Bayen, M. Annavaram, and Q. Jacobson. Virtual trip lines for distributed privacy-preserving traffic monitoring. In *ACM MobiSys*, pages 15–28, 2008.

[26] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Enhancing security and privacy in traffic-monitoring systems. *IEEE Pervasive Computing*, 5(4):38–46, 2006.

[27] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Preserving privacy in gps traces via uncertainty-aware path cloaking. In *ACM CCS*, October 2007.

[28] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. K. Miu, E. Shih, H. Balakrishnan, and S. Madden. CarTel: A Distributed Mobile Sensor Computing System. In *ACM SenSys*, 2006.

[29] T. Jiang, H. Wang, and Y.-C. Hu. Preserving location privacy in wireless lans. In *ACM MobiSys*, 2007.

[30] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE TKDE*, 19(12):1719–1733, 2007.

[31] J. Krumm. Inference attacks on location tracks. In *Pervasive*, 2007.

[32] M. Lighthill and G. Whitham. On kinematic waves. II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 229(1178):317–345, 1955.

[33] J. Lite. Obama's cell phone hacked, privacy issues murky. http://www.scientificamerican.com/blog/post.cfm?id=obamas-cell-phone-hacked-privacy-is-2008-11-21.

[34] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *IEEE ICDE*, page 24, 2006.

[35] T. Park and S. Lee. A Bayesian approach for estimating link travel time on urban arterial road network. In *ICCSA*, pages 1017–1025, 2004.

[36] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, Dec 1979.

[37] P. I. Richards. Shock waves on the highway. *Operations Research*, 4(1):42–51, 1956.

[38] S. Saroiu and A. Wolman. I am a sensor, and i approve this message. In *ACM HotMobile*, pages 31-36, 2010.

[39] A. Skabardonis and N. Geroliminis. Real-time estimation of travel times on signalized arterials. In *Proceedings of the 16th International Symposium on Transportation and Traffic Theory*, University of Maryland, College Park, MD, July 2005.

[40] L. Sweeney. Achieving k-Anonymity Privacy Protection Using Generalization and Suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):571–588, 2002.

[41] B. Wiedersheim, Z. Ma, F. Kargl, and P. Papadimitratos. Privacy in inter-vehicular networks: why simple pseudonym change is not enough. In *IEEE WONS*, pages 176–183, 2010.

[42] D. B. Work, S. Blandin, O.-P. Tossavainen, B. Piccoli, and A. M. Bayen. A Traffic Model for Velocity Data Assimilation. *Applied Mathematics Research eXpress*, 2010(1):1–35, 2010.