Making Connected Cars Untraceable via DSRC Radios

JAEMIN LIM, KIYEON KIM, HYUNWOO YU, SUK-BOK LEE, (Member, IEEE)
Department of Computer Science and Engineering, Major in Bio Artificial Intelligence, Hanyang University, Ansan 426-791, South Korea
Corresponding author: Suk-Bok Lee (e-mail: sble@hanyang.ac.kr).
This work was supported by the research fund of Hanyang University (HY-2014-N)

ABSTRACT This paper shows potential of using DSRC radios for vehicle tracking protection. We focus on traffic data collection as our target application where vehicles send their locations to a server while driving. Such vehicles are easily trackable—revealing location history—as the application often requires frequent and accurate location updates.

This paper presents PathCloak, a method that enables vehicles to report their locations while preventing the server from properly tracking the vehicles. PathCloak leverages vehicles’ two network interfaces: in-car Internet (for accessing the server) and car-to-car DSRC (for creating path confusion). PathCloak-enabled vehicles exchange their kinematic information via DSRC radios, and use it to generate plausible path segments for each other, making their paths indistinguishable each other from the server’s viewpoint. We demonstrate its feasibility via field experiments on real roads using our DSRC testbeds. Our evaluation shows that PathCloak offers strong privacy (tracking success ratio<1%), while maintaining high utility for various traffic statistics.

INDEX TERMS Smart cars, tracking protection, utility preservation, DSRC radios

I. INTRODUCTION
Dedicated short-range communication (DSRC) is a vehicular communication technology that has long been a pilot service worldwide [1]–[4]. These long-term tests showed that DSRC can improve road safety and reduce traffic accidents. The key instrument is the basic safety message, a DSRC beacon that contains vehicle position, heading, speed, and other information about a vehicle’s state and predicted path. Surrounding vehicles receive the message, and each estimates the risk, e.g., collision threats, imposed by the transmitting vehicle. Such enhancement of road safety has long been tested as the first prominent use of DSRC on the roads.

In this work, we explore new potential of DSRC that can enhance vehicle privacy using existing DSRC beacons. We focus on traffic data collection as our target application where vehicles send their locations to a central server while driving. Such vehicles are easily trackable as the application often requires frequent and accurate location updates. Specifically, the server that collects periodic user locations could use them to track the vehicles and come up with their location history. Even with anonymized location reports, the server-side time-series analysis on such location samples (i.e., following the footstep)—also combined with personally identifiable position data—could accumulate path information and eventually identify users’ location history.

This paper presents PathCloak, a method that enables vehicles to report their locations while preventing the server from properly tracking the vehicles. PathCloak leverages vehicles’ two network interfaces: in-car Internet and car-to-car DSRC. Vehicles send locations to the server via in-car Internet while protecting against tracking via car-to-car DSRC. To this end, we utilize moving vehicles’ kinematic information—position, speed, heading, etc—that is originally contained in a regular DSRC beacon (broadcasted 10 times per second). By exchanging such information via DSRC radios, vehicles generate plausible, mutually-obfuscating path segments for each other. Each path segment is created in a way that it diverges from one’s actual path and joins the other’s actual path. Such a PathCloak operation is brief and can thus be performed multiple times during the trip. As a result, a vehicle’s path keeps branching off from the viewpoint of observers.

PathCloak has several merits: (i) vehicles send their actual updates all the time, ensuring spatial-temporal accuracy; (ii) car-to-car DSRC (range of 300–400m) enables a flexible form of path confusion between vehicles moving in various
allowed to track users’ activities with users’ consent. Identities cannot be used anonymously. Such an application is implicitly aimed to capture data that accurately reflects the real-world traffic situations, and offers strong privacy (tracking success ratio < 1%) while retaining high utility for various traffic statistics such as congestion estimation, traffic flow modeling, etc.

It is worth mentioning that the protection performance of PathCloak depends on the DSRC-equipped rate. In other words, the more vehicles are equipped with DSRC radios, the more PathCloak operations are likely to occur during a trip—the greater the confusion. Thus, a high equipped rate is desired for PathCloak. We do not expect PathCloak to be an immediate rollout as the standard has not been fully adopted by vehicle manufacturers. In this work, we intend to report the new potential of DSRC via PathCloak as it is anticipated that cellular-based in-car Internet and DSRC will be deployed in coexistence as essential features for soon-to-come connected cars [5]–[8].

In summary, we make the following contributions:

- We unlock the potential of DSRC to protect location privacy. This opens up a new opportunity for DSRC, which was previously used predominantly for road safety.
- We build a full-fledged prototype and conduct real road experiments using PathCloak-enabled vehicles. Our field measurement provides insights about necessary conditions for vehicles to perform PathCloak operations in reality.
- We examine PathCloak-generated traces from both privacy and utility perspectives. We evaluate the privacy using the most advanced deep-learning based tracking. We assess the utility using three different traffic data analysis tasks.

II. MOTIVATION

A. TRAFFIC DATA COLLECTION

Our work specifically focuses on location reporting applications that do not explicitly require a user identity.1 In this structure, users only send their GPS coordinates to the server while driving. This type of applications is intended to collect users’ location data for various traffic statistics. They range from traffic monitoring [9]–[11], to traffic congestion assessment [12]–[14], to mobility pattern estimation [15]–[18], to dashcam data reporting [19]–[21]. These applications aim to capture data that accurately reflects the real-world traffic situation. These applications aim to capture data that accurately reflects the real-world traffic situation. Spatial-temporal accuracy of user reports as well as their reporting frequency thus play an important role for the quality of data accumulated in the server. For this reason, some fine-grained, crowdsourced monitoring systems have their user reporting as frequent as every second [19].

These location reports can be made anonymous via a simple practice: (i) users can hide network identifiers via anonymization networks like Tor or I2P; and (ii) users can also constantly change their externally-visible network IDs (e.g., by switching Tor circuits). This will prevent network IDs from being used for distinguishing among users. However, such anonymized reports do not guarantee privacy as below.

B. THREAT MODEL

Our primary adversary in this work is a hostile server or anyone with access to the database. These attackers may wish to track users via the collected location samples. Especially, if a user sends location reports frequently (even with constantly changing network IDs), this creates a trail of his/her location, allowing a hostile server to easily follow a user’s path.

Such location tracking can further reveal users’ very private information if the server can connect specific individuals to specific locations [22]–[24]. For example, an anonymous user at a hospital or other private location can be eventually identified if the user’s path is tracked from his/her home (as a resident of a particular address). We assume a strong adversary with the combination of such identification and tracking capabilities by cross-referencing the coordinates of the location reports with a database, and using powerful tracking techniques, e.g., state-of-the-art machine-learning based tools to reconstruct a target user’s path from anonymized location samples.

C. GOAL MODEL

Tracking protection. A collection server should not be able to trace a user’s path. More specifically, given a start position of a user, the server should not be able to determine an end position of the trip, and vice versa. Such traceability should be retained even under frequent and high-accuracy location updates.

Utility preservation. Enabling tracking protection should not compromise the quality of collected statistics. For example, reporting too much fake location data will result in an excessive noisy database, making it suffering from serious utility losses. The servers should be able to maintain utility-preserving, fine-quality databases.

D. EXISTING APPROACHES AGAINST TRACKING

Space-time intersection. Mix-zones [25]–[29] make users’ paths indistinguishable if they coincide in space and time. As shown in Fig. 1a, one cannot determine whether the cars turned or went straight. However, such space-time inter-
sections (i.e., two vehicles occupy exactly the same place at the same time) are not common in practice, especially under frequent and high-accuracy location updates.

**False trips.** A user can create false trips by reporting false locations along with the real one [30]–[33]. However, it requires that false reports vastly outnumber the real ones. As in Fig. 1b, making the chance of an attacker picking the true path below 0.01 requires the individual user to report at least 100 different false paths. The more users adopt this approach, the more overwhelming the volume of “noisy” data accumulated in the system, severely reducing utility.

**Differentially-private traces.** Data collectors can publish trajectory data in the form of synthesized location traces that achieve differential privacy [34]–[36]. A recent work [36] further presents a high-utility trace synthesizer with differential privacy guarantees. However, they are inherently server-driven privacy techniques where data collectors have raw user trajectory data and process it for publication. User location history is known to the data collector.

### III. PATHCLOAK: COOPERATIVE PRIVACY

As mentioned earlier, false trips can create confusion but will incur serious utility losses. The main culprit is that each individual user produces a large number of false paths for his/her own protection (Fig. 1b). We suggest using inter-vehicle communication to do a cooperative form of tracking protection with minimal false data. With cooperation between vehicles (via DSRC radios), we can create confusion with pairs of short false paths (or false pathlets) to collectively realize the full-trajectory protection, as illustrated in Fig. 1c.

**A. PATHCLOAK OVERVIEW**

Here we briefly explain how PathCloak works (see Fig. 2). When vehicles $A$ and $B$, both reporting to server $S$, hear each other’s DSRC beacon (range of 200–400m), they determine whether to generate false pathlets for each other. If yes, from this moment ($t_0$ in Fig. 2a), each vehicle $A$ independently sends $S$ two location reports (via two new network IDs): one is $A$’s actual location; and the other is $B$’s false location. Once the actual and false pathlets converge (e.g., $t_0 + \alpha$ in Fig. 2b), the vehicle sends $S$ only its actual locations (via a new network ID) and performs, whenever possible, the PathCloak process with others as it moves on.

**Outcome of PathCloak.** PathCloak achieves the followings: (i) vehicles send their actual location reports all the time (spatial-temporal accuracy); (ii) the number of false reports during a trip is strictly less than actual ones (data quality); and (iii) continuously divergent paths prevent the server from determining vehicles’ actual trajectories (privacy).

**Compared with Mix-zones.** Mix-zones create confusion about vehicle paths with help from RSUs (roadside units). However, it is specifically designed to protect against a passive adversary who installs its own, globally-placed receivers in roadides and tries to eavesdrop vehicle location information in DSRC beacons. In other words, mix-zones are not intended (thus unable) to provide privacy under location-reporting applications where vehicles continuously send their locations to a remote server. Whereas, PathCloak is tailored for such applications, protecting against tracking via inter-vehicle cooperation even without help from RSUs.

**Why cooperative?** PathCloak ensures privacy only when vehicles generate false pathlets for each other. A selfish user may not collaborate; the user may not produce a false pathlet for the other who does for him/her. We, however, argue that such uncooperative behavior rather threatens his/her own privacy. Suppose that $B$ in Fig. 2 does not generate a false pathlet for $A$. Then, the server could easily recognize $B$’s actual path via a posteriori reasoning. Without $A$’s false pathlet (created by $B$), there will be only a single path from $A$’s position at time $t_0$ to $t_0 + \alpha$. This will negate $B$’s false pathlet (created by $A$), and thus $B$’s actual path will be identified. Such mutually beneficial condition in PathCloak naturally encourages user participation.
B. PATHCLOAK OPERATION

1) Pairing Between Vehicles

Each vehicle $A$ determines whether to perform PathCloak with other vehicle $B$ based on received DSRC beacons, especially $B$’s current location $\text{loc}^B_{\text{cur}}$ and speed $\text{speed}^B_{\text{cur}}$. $A$ assesses whether its actual path and $B$’s false pathlet could converge in the predefined time period $T_f$ (without significantly changing $B$’s speed) as follows:

$$\frac{\text{Dist}(\text{loc}^B_{\text{cur}} \rightarrow \text{loc}^A_{\text{T_f}})}{\text{speed}^B_{\text{cur}} \times (1 + \lambda)} < T_f < \frac{\text{Dist}(\text{loc}^B_{\text{cur}} \rightarrow \text{loc}^A_{\text{T_f}})}{\text{speed}^B_{\text{cur}} \times (1 - \lambda)}$$

where $\text{loc}^A_{\text{T_f}}$ is $A$’s estimated future position (in $T_f$) based on $A$’s current speed and route ahead provided by in-car navigation system. $\text{Dist}(\text{loc}^B_{\text{cur}} \rightarrow \text{loc}^A_{\text{T_f}})$ is a driving distance\(^2\) from the start point of $B$’s false pathlet ($\text{loc}^B_{\text{cur}}$) to the provisional convergence point ($\text{loc}^A_{\text{T_f}}$). $\lambda$ is the predefined margin that limits the short-term speed change on the false pathlet to avoid trackers’ suspicion. Note, $A$’s exact future positions can also vary from the estimated ones due to its speed change. Thus, satisfying Inequality (1) does not pinpoint the convergence time, but means that $A$’s actual path and $B$’s false pathlet are likely to converge in about $T_f$.

If the beacon from $B$ satisfies Inequality (1), $A$ sends $B$ a propose message. Likewise, $B$ responds with an accept message if confirming the joinability of $B$’s actual path and $A$’s false pathlet based on $A$’s beacon. Note that these PathCloak messages are contained in a regular DSRC beacon and thus are broadcasted 10 times per second. If no accept message from $B$ is received within one second, then $A$ considers that this proposal is unsuccessful and proceeds with other candidates as moving on. One second of waiting is chosen because it gives enough time to deal with possible losses of DSRC beacons (i.e., propose/accept messages) while not disrupting the periodic (second-scale) location reporting to the server. It is also worth mentioning that, despite several retransmissions within one-second waiting time, a permanent loss of an accept message, if ever, can cause a “half-pairing”; the one who sent the accept message creates a false pathlet while the proposer does not. We, however, point out that such a case is not only rare but also tolerable without permanent disruption of the full-trajectory protection as the vehicles continue to perform, whenever possible, the PathCloak operations with others as they move on.

Once a mutual pairing is made, each vehicle starts to independently send the server two location reports (via two new network IDs): actual (for itself) and false location (for the other). We here use Tor to switch vehicles’ network IDs. Whenever new network IDs are needed, the vehicle use new Tor circuits. To do so, the vehicle prepares new circuits in advance while driving, as it takes 5–7 secs to build a new circuit. Fig. 3 shows an example of such location reporting from the network-ID perspective of vehicle $A$. From this moment when the vehicles get paired, their DSRC beacons contain a pairing-on message, indicating that they are not currently available for pairing with others. Fig. 4 shows the finite state machine that specifies a PathCloak-enabled vehicle.

2) Plausible Pathlet Generation

We describe the false pathlet generation by vehicle $A$, and it is a reciprocal procedure for $B$.

**Convergence point selection.** A first chooses the convergence time $T_{\text{conv}}$. $A$ picks $T_{\text{conv}}$ randomly within Inequality (1) by replacing $T_f$ with $T_{\text{conv}}$ to fix the convergence position $L_{\text{conv}} (= \text{loc}^A_{\text{T_{\text{conv}}}})$ in advance. That is, $A$ will reach $L_{\text{conv}}$ if moving along the route with default speed $\text{speed}^A_{\text{dflt}} (= \text{speed}^A_{\text{dflt}})$ for $T_{\text{conv}}$. And, $B$’s false position will be at $L_{\text{conv}}$ if moving with default speed $\text{speed}^B_{\text{dflt}}$ for $T_{\text{conv}}$ as follows:

$$\text{speed}^B_{\text{dflt}} = \frac{\text{Dist}(\text{loc}^B_{\text{cur}} \rightarrow \text{loc}^A_{\text{T_{\text{conv}}}})}{T_{\text{conv}}}$$

Note that resulting $\text{speed}^B_{\text{dflt}}$ is within the margin $\lambda$ of $B$’s current speed $\text{speed}^B_{\text{cur}}$ as $\lambda$ is already reflected in $T_{\text{conv}}$. Now $A$ is ready to generate $B$’s false location reports. **Reporting locations.** The key idea is that, while $A$ may change its speed, keeping the ratio of $\text{speed}^A_{\text{dflt}} : \text{speed}^B_{\text{dflt}}$ (by adjusting fake $B$’s speed accordingly) will lead them to convergence location $L_{\text{conv}}$ at the same time (not necessarily

Fig. 3: PathCloak operation example from the net-ID perspective of vehicle $A$ when reporting locations to the server.

Fig. 4: Finite state machine for a PathCloak-enabled vehicle.
at $T_{conv}$). More specifically, on $i^{th}$ reporting time where $A$’s actual location is $loc^A_i$, $B$’s false position $loc^B_i$ is created via the following proportionality: $\text{dist}(loc^A_{i-1} \rightarrow loc^A_i) : \text{dist}(loc^B_{i-1} \rightarrow loc^B_i) = speed^A_t : speed^B_t$ where $\text{dist}(\cdot)$ is a moving distance since the last reporting time. Note that $B$’s false locations are placed along the route of $\text{Dist}(loc^A_{i_{\text{curr}}} \rightarrow L_{conv})$ obtained (in Eq. (2)) by in-car navigation system. The resulting false pathlet reflects such a realistic route and has its coordinates variably spaced along the route. $A$ sends $loc^A_i$ and $loc^B_i$ via two different network IDs. Once the actual and false pathlets converge, i.e., when $A$ reaches the convergence location $L_{conv}$, $A$ only sends the actual locations over a new network ID. From this moment, $A$’s DSRC beacons contain a pairing-off message, indicating that it is available for pairing with others.

IV. THE COST OF PATHCLOAK

Communication. The format of a PathCloak message is as follows: message type (2 bits: propose, accept, pairing-on, or pairing-off), counterpart vehicle ID (6 bytes), and an LBS identifier (4 bytes).\(^3\) The length of our PathCloak message is thus only 11 bytes. It can be piggybacked into a DSRC beacon—Basic Safety Message (BSM) whose size is as large as nearly 300 bytes [37], as shown in Fig. 5.

![Fig. 5: DSRC basic safety message (BSM) structure.](image)

False data. A PathCloak-enabled vehicle generates false data only when being paired for the operation. Thus, its volume never exceeds that of the real one. Our result from the PathCloak experiment (in Section VI) shows that the total volume of false updates is typically 15–20%. Note that, in the same settings, independent false trips [30]–[33] would incur at least 100 times more number of false updates than PathCloak in order to achieve the same degree of privacy.

Computation. Three key procedures that require computation are: (1) pathlet joinability check (when pairing); (2) convergence point selection (upon paired); and (3) false location creation (until pairing-off). Table 1 shows the time taken in each procedure when running our PathCloak implementation on a Raspberry Pi as well as other platforms. Proc (1) takes more time than Proc (2) and (3) as it runs in-car navigation API—we use GraphHopper directions API [38]—to compute driving distances of false pathlets. Still, the processing time on a Raspberry Pi is small enough to fit in the DSRC beacon interval of 100 ms. Note, in-vehicle computers today have higher capabilities than Raspberry Pi [39]. We expect that it runs even faster in reality.

V. EXPERIMENTS ON REAL ROADS

A. MEASUREMENT FRAMEWORK

Our field measurement answers the following questions:

- On what condition do vehicles get paired and perform PathCloak operations in reality?
- How plausible are the false pathlets that PathCloak produces from real-world driving?

Prototype implementation. Our testbed consists of on-board DSRC units (deployed in a rear window), Raspberry Pis with LTE module (for in-car Internet access), and 7” touchscreen (for display) as depicted in Fig. 6. We implement PathCloak on Raspberry Pis, which connects (via an Ethernet cable) to the DSRC OBU for inter-vehicle PathCloak-message exchange. For in-car navigation, we use GraphHopper [38], an open-source, fast directions API to set actual routes ahead and to compute false pathlets. All our source code is available at https://github.com/inclincs/pathcloak.

Experiment settings. We conduct our experiments for six weeks in downtown/residential areas of Seoul, Korea. We run experiments using two vehicles equipped with our PathCloak-enabled DSRC testbeds in various situations. The vehicles—we vary their locations, moving directions, and speeds—send out PathCloak messages via 802.11p-based DSRC broadcast. These messages are contained in a regular DSRC beacon and thus are broadcasted 10 times per second. The transmission power is set to 14 dBm as recommended in [37]. In our field experiments, we use $\lambda = 0.3$ and $T_f = 30$ secs—which are the recommended values and will be discussed in Section VI. We make the vehicles (as users) send their location reports every second via the LTE interface to our collection server.

B. MEASUREMENT RESULTS

In the field tests, we use the following two metrics:

- Pairing success rate (P-rate): Ratio of the number of runs where the two vehicles get paired and perform PathCloak operations to the total number of runs.

| Platform          | Check time | Select time | Create time |
|-------------------|------------|-------------|-------------|
| Rasp. Pi 3 (1.2 GHz) | 30.1 ms    | 9.3 ms      | 0.12 ms     |
| MacBook (2.4 GHz)  | 11.7 ms    | 2.1 ms      | 0.03 ms     |
| MacBook (4.0 GHz)  | 7.5 ms     | 0.9 ms      | 0.01 ms     |

Table 1: Processing time of the PathCloak operation.

\[^3\text{An LBS identifier specifies the vehicle’s currently-used location-reporting application. Note that only the vehicles with the same LSB id can get paired. This ensures that PathCloak operations will be performed by the vehicles reporting to the same server(s).}\]
• Tracker’s uncertainty: Normalized entropy of probability that the attacker correctly picks an actual pathlet upon each individual PathCloak operation.

**Experiment-Set 1: Vehicles Are Making Right Turns**

The first experiment-set reflects a situation where vehicles make right turns in opposite directions (Fig. 9). By varying the vehicle speeds, we conduct a set of four different experiments, each of which is carried out over 30 times. The vehicle speeds of these four cases are: 
1. 40:40km/h; 
2. 40:50km/h; 
3. 30:40km/h; and 
4. 30:50km/h. Note, the speed limit in downtown/residential areas of Seoul is 50–60km/h [40], and our experiments have obeyed this speed limit. We make our vehicles (at least 800m apart) move from opposite sides of the intersection. When making a turn, our vehicles have temporarily reduced the speeds accordingly.

![Fig. 9: [Ex-Set1] Actual and false pathlets.](image)

Fig. 7 shows the result of pairing success rate (P-rate) from these experiments. We get high P-rate (＞ 83%) in the first three cases and the reduced P-rate (47%) in the last case (30:50km/h). This suggests that the speed difference between vehicles impacts the P-rate. The larger the speed difference, the greater the disparity in lengths of the corresponding actual and false pathlets (due to the greater difference in their estimated moving distances in $T_f$; see Eq. (1)), making it hard to satisfy the PathCloak triggering condition.

![Fig. 7: [Ex-Set1] Pairing success rate.](image)

Table 2 shows the average vehicle distance at the time when PathCloak is triggered. The pairing range is as far as 430m. We also observe that, throughout the experiments, PathCloak operations do not require the paired vehicles to reach the intersection at the same time. This shows that PathCloak indeed works without space-time intersection of vehicles. Fig. 9 depicts sample pathlets from the experiment.

**Table 2: [Ex-Set1] Pairing distance.**

| Vehicle speed (car1 : car2) | Average pairing distance | Max. pairing distance |
|-----------------------------|--------------------------|-----------------------|
| 40 : 40km/h                 | 207m                     | 437m                  |
| 40 : 50km/h                 | 282m                     | 414m                  |
| 30 : 40km/h                 | 321m                     | 429m                  |
| 30 : 50km/h                 | 286m                     | 384m                  |

**Experiment-Set 2: One Moving Straight, The Other Turning Right**

The second experiment-set reflects a situation where one vehicle moves straight while the other makes a right turn (both initially in the same direction), as depicted in Fig. 10. We again conduct a set of four experiments but with different vehicle speeds as follows: 
1. 40:40km/h; 
2. 40:50km/h; 
3. 30:50km/h; and 
4. 20:40km/h. In each experiment, we make the slower vehicle take a right turn at the intersection.

![Fig. 10: [Ex-Set2] Actual and false pathlets.](image)

Fig. 8 shows the result from the experiments. We get high P-rate (＞ 75%) in the first three cases—even for the third one (30:50km/h). Whereas, only 17% P-rate is attained in the last case (20:40km/h). This result indicates that, besides the speed difference between vehicles, their actual speeds also affect the P-rate. When a vehicle move at a slower speed, the corresponding PathCloak triggering condition becomes tighter (Eq. (1)), making it hard to get paired with others.

**Experiment-Set 3: Vehicles Are Moving Straight in the Same Direction**

The third experiment-set represents a situation where both vehicles move straight in the same direction. This time, we make both vehicles move at the same speed—which is typical for vehicles in the same road. Instead, we vary the vehicle distance: 50m, 100m, 150m, and 200m.

![Fig. 11: [Ex-Set3] Actual and false pathlets.](image)

Fig. 12 and 13 show the results when both vehicles move at 35km/h and 45km/h, respectively. We get high P-rate (＞ 90%) within 100m. Whereas, no pairing is made in the...
longer-distance cases (except 150m at 45km/h). The reason for the short pairing distance (cf. 300–400m in the other scenarios) is that, in this setting, the vehicle distance is directly added (subtracted) to the lengths of false pathlets, making it a limiting factor for the triggering condition. Fig. 11 depicts sample pathlets from the experiment.

At first glance, this seems to offer little privacy as both vehicles still move in the same direction. However, point out that every single PathCloak operation (including such lane-level confusion) collectively contributes to full trajectory tracking protection in that: (i) after the confusion, the tracker of a target vehicle needs to follow both vehicles; (ii) their eventual destinations are not likely the same—their paths will diverge afterwards; and (iii) each of them continues to perform PathCloak with others—doubling the degree of confusion. Note, each PathCloak operation serves as a building block to collectively realize the full-trajectory protection.

Experiment-Set 4: Vehicles Are Moving Straight in Opposite Directions
This set of experiments represents a situation where vehicles move straight from opposite sides of the intersection. We vary the vehicle speeds: (i) 40:40km/h; (ii) 40:50km/h; (iii) 30:40km/h; and (iv) 30:50km/h.

Fig. 14 shows high P-rate (>78%) in the first three cases and low P-rate (26%) in the last case, which signifies the impact of relative vehicle speed. Note, this result is only valid for intersections that allow U-turns. We have also conducted the same set of experiments in the roads where U-turns are not permitted, and found that pairing is not possible—navigation does not provide false pathlets with U-turns in such roads. This indicates that traffic rules also affect the triggering condition in this situation. Fig. 18 depicts sample pathlets from the experiment.

Experiment-Set 5: Initially in Opposite Directions. One Moving Straight, The Other Turning Right
This represents a situation where one vehicle moves straight while the other makes a right turn (initially in opposite directions), as depicted in Fig. 19.

Fig. 15 shows high P-rate (>78%) in the first three cases and low P-rate (23%) in the last case—a similar trend to Ex-Set 4. Note, we were able to obtain this result only in the intersections with U-turns. This signifies that, if either one of the vehicles moves straight in the opposite direction, the key requirement for pairing them is the allowance of U-turns.

Experiment-Set 6: One Moving Straight, The Other Turning Left
This reflects a situation where one vehicle moves straight while the other makes a left turn, as depicted in Fig. 20.

Fig. 16 shows high P-rate in the first three cases and the reduced P-rate (58%) in the last case, displaying again the effect of relative vehicle speed. Note that many intersections
in Korea allow going straight and (protected) left turns at the same time. This set of experiments has been performed under such traffic light control.

Nevertheless, the above results are still valid for other types of intersections. Unlike Ex-Set 5, this situation does not require specific traffic rules (e.g., U-turns) in the creation of false pathlets. Thus, the speed difference is the major factor in this scenario. For example, in the case of unprotected left-turns, the vehicles are likely to move much slower (or even stop) as reaching the intersection while straight-moving vehicles pass the intersection without slowing down. Such a speed difference would correspond to our last case (30:50km/h) or worse—hence the lower P-rate.

**Experiment-Set 7: One Turning Right, The Other Turning Left**

This represents a situation where one vehicle makes a right turn while the other takes a left turn, as depicted in Fig. 21.

Fig. 17 shows high P-rate in the first two cases, whereas P-rate of 43% at 30:50km/h and 12% at 20:40km/h. This reconfirms that both the relative and the actual vehicle speeds are the key factors for the triggering condition. Note that the above results are also applicable to any types of intersections for the same reason as Ex-Set 6.

**Experiment-Set 8: Vehicles Are Approaching from Perpendicular Directions**

This experiment-set reflects a situation where vehicles approach an intersection from perpendicular directions. In this situation, large buildings at the intersection corners, if any, may restrict the DSRC communication between the vehicles—the wireless signal could be blocked by the buildings. To assess its impact on the P-rate, we conduct the experiments in two different environments: line-of-sight (LOS) and non line-of-sight (NLOS).

Fig. 22a and 22b show the P-rate results in the LOS (Fig. 23a) and the NLOS intersections (Fig. 23a), respectively. In the LOS case, we get high P-rate (> 75%) when the vehicles move at the speeds of 40:40km/h and 40:50km/h—a similar trend to Ex-Set 7. Whereas, in the NLOS case, the P-rate decreases considerably—only 41% successful pairing even at the speeds of 40:40km/h. We observe that, in the NLOS intersection, pairing is possible only at the time when one vehicle passes the corner of the building towards the intersection (e.g., vehicle A in Fig. 24), reaching the LOS position to the other. Thus, the pairing condition in an NLOS intersection is restricted to a specific timing—just prior to entering the intersection. This rules out pairing opportunities that might exist at other timings if not with the DSRC blockage.

Our result indicates that, when vehicles approach from perpendicular directions, large obstacles (e.g., buildings) around the intersection have significant impact on the likelihood of their pairing. This observation also conforms to the previous empirical and theoretical results reported in [41]–[43]. We thus expect that such “perpendicular” pairing is more likely to happen in residential or suburban areas rather than downtown areas with many tall buildings.

**Plausibility Experiment: Tracker’s Uncertainty**

To examine the plausibility of false pathlets created from our experiments, we measure the tracker’s uncertainty in actual
pathlet prediction. For the tracker prediction, we use the ST-RNN model (see Section VI). As there is no publicly available real-world driving dataset in Seoul, Korea, we collect traces by ourselves. Using our vehicles, we record their GPS coordinates every second while driving in the area where we have conducted the PathCloak experiments. Our two-week-long effort provides a dataset of 1K trajectories5 that capture various moving patterns in the area. We use them to train the ST-RNN model for pathlet prediction.

Table 3 shows the tracker’s uncertainty when applying the prediction to PathCloak traces from the previous Ex-Sets 1–8. We use the normalized entropy as the uncertainty index and thus, it is valued between 0 and 1. When the tracker is fairly unsure which pathlet is actual, the resultant index is close to 1. Our result above shows that the false pathlets from real-world driving are plausible enough to confuse the tracker.

| Experiment | Tracker’s uncertainty |
|------------|-----------------------|
| Set 1      | 0.998                 |
| Set 2      | 0.993                 |
| Set 3      | 0.997                 |
| Set 4      | 0.984                 |
| Set 5      | 0.985                 |
| Set 6      | 0.988                 |
| Set 7      | 0.996                 |
| Set 8      | 0.993                 |

**TABLE 3:** Tracker’s uncertainty in prediction.

We use the normalized entropy as the uncertainty index and (ii) people tend to drive near the speed limit (rather than driving slow). The plausibility result is also positive. Given a PathCloak operation, the tracker has high uncertainty about which pathlet is actual. This demonstrates that each PathCloak operation generates highly plausible pathlets, successfully creating en-route confusion on real roads.

**Summary of Measurement Results**

The results of our field tests show that PathCloak works well (P-rate>80%) in many cases. Our general observation is that P-rate between cars becomes high when: (i) the speed difference is small; and/or (ii) their actual speeds are high. This has practical implications for real-world drivers because: (i) nearby cars usually move at similar speeds; and (ii) people tend to drive near the speed limit (rather than driving slow). The plausibility result is also positive. Given a PathCloak operation, the tracker has high uncertainty about which pathlet is actual. This demonstrates that each PathCloak operation generates highly plausible pathlets, successfully creating en-route confusion on real roads.

**VI. THE PRIVACY OF PATHCLOAK**

We now evaluate PathCloak in a large-scale environment, focusing on its full trajectory protection. Here we consider the collection servers as potential attackers with tracking capabilities. We experiment with two prominent tracking methods: Deep-learning based and Markov-based approaches.

**A. ATTACKER’S TRACKING MODEL**

**Deep-learning based tracking.** We use a spatial-temporal recurrent neural network (ST-RNN) model [44], [45], one of the most-accurate location prediction methods today. We build it using Keras [46], a neural-networks API running on top of TensorFlow [47]. We obtain the GeoLife dataset [48]–[50], a real-world location history dataset from Beijing, China, containing five-year driving records of 182 users (+17K trajectories) with their GPS logged every 1–5 seconds. We use them as training input to the ST-RNN model that internally performs learning and captures spatio-temporal transition regularities for future location prediction.

**Markov-based tracking.** We also test with a Markov-based model, which is traditionally widely used for mobility prediction [51]–[53]. Using the GeoLife dataset as a historical data, we generate a Markov transition matrix over a grid of blocks in a target area. For vehicle tracking, when a PathCloak operation is triggered, the attacker determines which one belongs to the actual path by predicting the most probable next block using the Markov matrix.

**B. PRIVACY METRICS**

**Location entropy.** We measure the degree of privacy using location entropy, \( H_t = -\sum_i p(i,t) \log p(i,t) \), where \( p(i,t) \) is the attacker’s belief (probability) that location sample \( l(i,t) \) from anonymous user \( i \) of time \( t \) belongs to the vehicle tracked. Lower values indicate more certainty or lower privacy.

**Tracking success ratio.** To give more intuitive privacy results, we also assess a tracking success ratio \( S_t \), that measures the chance that the tracker’s belief, when tracking a target over time \( t \), is indeed true. Thus, \( S_t \) is equivalent to \( p(u,t) \) of actual target \( u \), since \( \sum_i p(i,t) = 1 \) at any time \( t \). Note, \( S_t \) is unknown to the tracker, who becomes unsure which \( l(i,t) \) belongs to target \( u \) over time.

**C. PRIVACY EXPERIMENTS**

We run PathCloak over the GeoLife dataset. To test with a less (more) number of participants, we experiment with 50–1000 vehicle trajectories in a certain 6×6km² area of Beijing. Using them as baseline mobility patterns over the road network (extracted via OpenStreetMap [54]), we obtain PathCloak-enabled location traces via ns-3 simulator.

**Choice of \( \lambda \) and \( T_f \).** We first assess the privacy implications of \( \lambda \) and \( T_f \), two key parameters for the PathCloak triggering condition (see Eq. (1)). To do so, we run PathCloak while varying their values.

**Fig. 34** shows the prediction success rate—the tracker’s probability of picking an actual pathlet upon a single PathCloak operation—with different vehicle speeds. We see that: (i) the larger the speed-change margin \( \lambda \), the less plausible the false pathlets; and (ii) such a trend becomes more significant in higher-speed roads. To have the value of \( \lambda \) as large as possible—more likely triggering a PathCloak operation—while limiting the tracker’s successful prediction, we choose to use \( \lambda=0.3 \) in urban roads and \( \lambda=0.2 \) in highways.

**Fig. 25, 28, and 31** show the location entropy of 20-min driving with different values of \( T_f \). We observe the diminishing returns in all the scenarios. This is, in fact, quite intuitive in that a small value of \( T_f \) restricts the triggering condition while a large value incurs a fewer number of PathCloak operations for a given trip time. To balance this tradeoff, we...
choose to set \( T_f = 30 \) seconds, which leads to the highest location entropy across all the scenarios.

**Low-density result.** Fig. 26 shows the average entropy with 50–150 users (1.3–4.2 users/1km\(^2\)). With 20-minute driving, vehicles reach four bits of location entropy even in the sparse case of \( n = 50 \). In other words, the tracker of a certain vehicle may suspect 16 different locations, but without knowing the exact location. Fig. 27 shows the tracking success ratio in the same setting. It decreases to 0.2 before seven minutes of driving and further drops below 0.1 before ten minutes.

**Medium-density result.** Fig. 29 shows the location entropy with 250–450 users (7.0–12.5 users/1km\(^2\)). In this more participatory setting, with 20-minute driving, a vehicle has more than six bits of entropy—the tracker’s confusion over at least 64 different locations—under the ST-RNN tracking. Note, the ST-RNN model consistently outperforms Markov-based tracking throughout the experiments. Fig. 27 shows the tracking success ratio in the same setting. It drops below 0.1 before seven minutes of driving.

**High-density result.** Fig. 32 shows the entropy with 600–1000 users (16.6–27.7 users/1km\(^2\)). With 20-min driving, vehicles reach more than seven bits of entropy, implying that the tracker of a certain vehicle may suspect at least 128 different locations not knowing exactly where to locate it. In this setting, the tracking success ratio drops below 0.1 before five minutes of driving (Fig. 33). Note that, without PathCloak, on the other hand, it still remains above 0.8 even after 20 minutes. This result shows: (i) the privacy risk from anonymous location data in its raw form; and (ii) privacy protection against tracking in the PathCloak-enabled traces.

### VII. THE UTILITY OF PATHCLOAK

We now assess PathCloak from a utility perspective. We measure how closely PathCloak-enabled traces reflect the real (raw) data for various traffic statistics.

#### A. UTILITY REFERENCE MODEL

We use the following prominent traffic-data analysis tasks to evaluate utility of PathCloak-enabled traces.
Traffic congestion estimation. Location updates from vehicles can provide realtime traffic information to predict traffic volume on road segments. We use the speed performance index [14] to estimate the road traffic state. The index value \( [0, 100] \) (percentage) is computed as a ratio between average vehicle speed and the maximum permissible road speed. Three threshold values \((25, 50, 75)\) are used for classification of road traffic level. The index values within \([0, 25), (25, 50), (50, 75), \) and \((75, 100)\) indicate road traffic level as heavy congestion, congestion, smooth, and very smooth, respectively.

Traffic mobility modeling. The collected vehicle trajectories also provide information about traffic dynamics in road networks. We use the Markov mobility model to capture vehicles’ intra-trajectory movement patterns. In doing so, we compute Markov transition matrix \( P_t \) and probability vector \( \pi_t \). Mobility modeling in such an aggregate level helps discover traffic flow patterns over road segments, benefiting road network planning, optimal allocation of roadside units, and provision of public transportation services.

Points of interest (PoI) extraction. The goal is to discover locations that are frequently visited and are prominently of interest to the public. The PoI extraction outputs the distribution of visits among locations, specifically the most visited locations—the top \( n \) popular places. This can have various commercial benefits, including ideal geographic placement for advertisements or retail stores, spatial hotspot discovery, and travel recommendations.

B. UTILITY MEASUREMENT

Analysis results from PathCloak-enabled traces may not be the same as those from the raw traces. We measure the difference between their results on the above analysis tasks.

Measurement settings. We extract 1000 vehicle trajectories from the GeoLife dataset and use them as our real traces. We obtain PathCloak-enabled traces over those raw trajectories in the road network. For comparison, we also generate traces using two other location-privacy preserving mechanisms (LPPMs): geo-indistinguishability (or geo-indi) and false trips. Geo-indi [55] is a generalization of differential privacy. It adds the planar Laplace noise to an actual location with parameter \( l = \varepsilon r \), where the user is protected within radius \( r \) with a privacy level \( l \) that increases with the distance. We set \( l = \ln(1.6) \) and \( r = 75 \) (meters). False trips [56] use many dummy routes. Each vehicle sends \( k \) false trips along with the actual one. Note, the parameter values for each LPPM are chosen to offer a similar level of privacy as PathCloak.\(^7\)

Traffic estimation result. We first assess the speed performance index (for each road segment) using the real data as well as the LPPM-processed data, and compare them. Fig. 35 shows the results—the cumulative distribution of the absolute value of their difference in speed performance index (between the real and the LPPM data). We see that the distribution of PathCloak leans toward zero, indicating that there are few “errors” in the traffic estimation. More specifically, the results are the same (or at most 5% difference) as those from the real data in more than 90% of the roads. On the other hand, for geo-indi and false trips, such accurate estimation happens in only 24% and 21% of the roads, respectively.

As mentioned earlier, traffic condition at each road segment can be classified into four congestion levels based on its speed performance index. Table 5 shows the resulting traffic levels. Across all the cases, the traffic levels from PathCloak data match those from the real data in more than 91% of the roads, indicating a high-level of preservation. On the other hand, we observe the significant discrepancies between the real and the estimated levels in the other two LPPMs. For example, 92.4% of the “Smooth” roads retain their actual level when estimated from PathCloak data, whereas only 34.8% and 30.8% of the roads do with geo-indi and false trips, respectively.

Mobility modeling result. We compare the mobility model \( P_t, \pi_t \) from the real traces and the model \( P'_t, \pi'_t \) from the LPPM-applied traces. To measure the difference between two mobility models, we use the earth mover’s distance (EMD) [57]. The EMD is widely used for evaluating dissimilarity between two probability distributions. We compute the expected EMD between two models: \( \mathbb{E}[d_{EMD}(P_t, \pi_t, P'_t, \pi'_t)] \) and \( \mathbb{E}[d_{EMD}(\pi_t, \pi'_t)] \) for each region \( i \) and time \( t \).

|               | PathCloak | Geo-indi | False trips | Uniform |
|---------------|-----------|----------|-------------|---------|
| \( P_t \)    | 2.0916    | 8.6335   | 14.3676     | 71.3358 |
| \( \pi_t \)  | 4.6663    | 3.0898   | 28.7593     | 30.3754 |

Table 4: Earth mover’s distance to the actual mobility model.

Table 4 presents the EMD results. Here, the uniform distribution is used as a baseline. We see that, for transition matrix \( P_t \) of each LPPM, the mobility model from the PathCloak traces is closest—most closely resembling—to the actual model from the real traces. As for \( \pi_t \), the model from the geo-indi traces has a smaller value, albeit little difference, than that from the PathCloak traces. This is because geo-indi does not produce any dummy location data unlike PathCloak and false trips. However, PathCloak minimizes the use of dummies and thus, its mobility model—even for \( \pi_t \) as well—closely reflects the actual model from the real traces.
Traffic state estimation from raw data

Traffic state estimation from LPPM-processed data

Table 5: Comparison of the road traffic estimation results from the LPPM data and from the real data.

Pol extraction result. For the Pol task, features of the most popular locations (i.e., the most frequently visited locations) are the most important ones to preserve. Across two datasets—the real and the LPPM, we measure the coverage \( \frac{n}{x} \), where \( x \) is the number of locations in their respective top \( n \) most frequently visited locations they have in common.

Table 6 shows the coverage results on the top \( n \) most frequently visited locations. We see that the coverage of the PathCloak traces is 100%. This is, in fact, quite an intuitive result. Such Pol-related information (e.g., \( n \) popular locations, etc) is retained entirely intact in the PathCloak traces because PathCloak never creates fake destinations—instead it creates en-route confusion using vehicles’ actual paths.

Overall, the statistics we have measured from the real traffic traces are well preserved in the PathCloak-enabled traces. Our evaluation results show that PathCloak is suitable for traffic data collection and traffic data analysis tasks via its privacy-preserving and high-utility trajectory data.

VIII. DISCUSSION

We discuss some remaining issues relating to PathCloak.

Use of multiple applications. Our PathCloak description so far focuses on cases where the vehicles use the same location reporting application. In reality, it is not uncommon to use multiple such applications at a time, not necessarily the same ones across users. This requires additional consideration because a PathCloak operation should be performed by vehicles reporting to the same server(s). In fact, this also applies to any existing methods that create obfuscation using other vehicles’ paths, such as mix-zones [25]–[29] and path confusion [58]. For this reason, they limit their description to the single application context.

Still, such methods can work under multiple applications via proper user matching. In our case, this requires a slight modification to the PathCloak message. By specifying a list of the currently-used location reporting applications (instead of a single app-id) in PathCloak messages, vehicles can get paired when they have any overlap in their lists. In this way, false pathlets from a single PathCloak operation can be simultaneously used for those overlapped applications. We believe such modification does not significantly increase the size of PathCloak messages, as individual users tend to use only a certain number of reporting applications at a time.

Realism of false pathlets. While our false pathlets reflect realistic routes, what remains unaddressed is traffic-signal awareness. At present, PathCloak does not take into account traffic signals when creating false pathlets. As a matter of fact, reflecting realtime traffic signals is one of the most challenging problems in false trip generation, and still largely unaddressed in the literature [30]–[33], [59]. One potential direction for PathCloak would be to use computer vision techniques to recognize the state of nearby traffic lights via in-vehicle cameras [60], [61] for creating false pathlets. Although not explored in this paper, we believe such traffic-signal awareness is necessary for increasing realism and merits separate research.

Navigation-assisted approach. The PathCloak operation requires the use of navigation systems in the driving process. In other words, we do not provide tracking protection for unequipped vehicles. We however expect that such vehicles will be uncommon in the near future. Indeed, in-car GPS navigation is becoming not only popular today, but also an essential feature for soon-to-come autonomous or self-driving cars (with auto-pilot features) that will enter mass production by 2022 [62].

IX. CONCLUSION

In this work, we report the new potential of DSRC via PathCloak that can protect vehicles from location tracking under traffic collection servers. The key insight is to leverage vehicles’ two network interfaces, in-car Internet (for accessing servers) and car-to-car DSRC (for obfuscating their paths). Our field experiments demonstrate that PathCloak-enabled vehicles (range up to 400m) successfully create path confusion for each other—by exchanging their kinematic information via existing DSRC beacons—under various real
road situations. The evaluation using real traffic traces shows that PathCloak provides strong privacy protection (tracking success ratio < 1%) while preserving high utility for the prominent traffic-data analysis tasks. In a broader scope, our solution explores to use untrusted location-collecting services in a privacy-preserving way while not giving up utility of the collected data statistics.

REFERENCES

[1] UK Centre for Connected Vehicles & Autonomous Vehicles. Multi-car collision avoidance report. Available online: https://nucca-project.co.uk/wp-content/uploads/2020/04/01294_MtUCCA-report.pdf (published in April 2020).
[2] US Department of Transportation. US DOT Safety Pilot Model Deployment Data 2020. Available online: https://catalog.data.gov/dataset/safety-pilot-model-deployment-data (published in Aug. 2020).
[3] Ministry of Transport South Korea. Intelligent transportation systems in Korea. Available online: http://www.molit.go.kr/upload/cyberJccr/pdf_file/ITS%２0brochure.pdf (published in May 2018).
[4] Society of Automotive Engineers of Japan. ITS standardization activities in Japan. Available online: https://www.jsae.or.jp/01info/org/its/2018_bro_e.pdf (published in Sep. 2018).
[5] K. Kielia, V. Barzdenas, M. Jurgo, V. Macaitis, J. Rafanavicius, A. Vasjanov, L. Kladovskis, and R. Navickas. Review of V2XÅSiTo standards and frameworks for ITS applications. In Journal of Applied Sciences, 2020.
[6] A. Costandou and M. Leba. Convergence of V2X communication systems and next generation networks. In Proc. International Conference on Applied Sciences, 2019.
[7] S. Chen, J. Hu, Y. Shi, L. Zhao, and W. Li. A vision of C-V2X Technologies, field testing, and challenges with Chinese development. In Journal of IEEE Internet of Things, 2020.
[8] E. Kim and Y. Shin. Feasibility analysis of LTE-based UAS navigation in deep urban areas and DSRC augmentation. In Journal of Sensors, 2019.
[9] TomTom. Real-time incident and road congestion analysis for traffic prediction. Available online: https://download.tomtom.com/open/banners/End-User-Traffic-Product-Sheet.pdf (published in 2019).
[10] HERE. Real-time traffic monitoring solution. https://www.here.com/.
[11] V. Verendel and S. Yeh. Measuring traffic in cities through a large-scale online platform. In Journal of Big Data Analytics in Transportation, 2019.
[12] F. Salma and S. Hosain. Enhancing mixed road traffic forecasting method using vehicle tracking system in GPS. In Proc. International Conference on Computing, Power and Communication Technologies, 2018.
[13] G. Kar, S. Jain, M. Gruteser, F. Bai, and R. Govindan. Real-time traffic estimation at vehicle road edge nodes. In Proc. ACM/IEEE Symposium on Edge Computing, 2017.
[14] Feifei He, Xuedong Yan, Yang Liu, and Lu Ma. A traffic congestion assessment method for urban road networks based on speed performance index. Procedia engineering, 137:425–433, 2016.
[15] Y. Zhao, X. Zhu, B. She, H. Yue, and M. Li. Exploring the weekly travel patterns of private vehicles using automatic vehicle identification data: A case study of wuhan china. In Journal of Sustainability, 2017.
[16] S. Zhang, J. Tang, H. Wang, Y. Wang, and S. An. Revealing intra-urban travel patterns and service ranges from taxi trajectories. In Journal of Transport Geography, 2017.
[17] D. Wang, Q. Liu, Z. Xiao, J. Chen, Y. Huang, and W. Chen. Understanding travel behavior of private cars via trajectory big data analysis in urban environments. In Proc. International Conference on Pervasive Intelligence and Computing, 2017.
[18] X. Liu, L. Gong, Y. Gong, and Y. Liu. Revealing travel patterns and city structure with taxi trip data. In Journal of Transport Geography, 2017.
[19] Minho Kim, Jaemin Lim, Hyunwoo Yu, Kiyeon Kim, Younghoon Kim, and Suk-Bok Lee. Viewmap: Sharing private in-vehicle dashcam videos. In Proc. USENIX NSLID, 2017.
[20] Sangkeun Park, Emilia-Stefania Ilincui, Jeungmin Oh, Sujin Kwon, Rabee Mitzoumi, and Uichin Lee. Facilitating pervasive community policing on the road with community policing system data. In ACM CHI, 2017.
[21] Dride Zero: Dashcam with cloud sharing. https://dride.io/features.
[22] C. Zhang and W. Wang. Vehicle trajectory reconstruction from global position system data with long short-term memory neural network. In Proc. International Conference on Urban Engineering and Management Science, 2020.
[23] Chris Y.T. Ma, David K.Y. Yau, Nung Kwan Yip, and Nageswara S.V. Raghavan. Privacy vulnerability of published anonymous mobility traces. In Proc. ACM MobiCom, 2010.
[24] Marco Gruteser and Xuan Liu. Protecting privacy in continuous location tracking applications. In IEEE Security and Privacy Magazine, 2004.
[25] A.R. Beresford and F. Stajano. Location privacy in pervasive computing. In IEEE Pervasive Computing, 2003.
[26] Julien Freudiger, Maxim Raya, Mark Felgehyázi, Panos Papadimitratos, and Jean-Pierre Hubaux. Mix-zones for location privacy in vehicular networks. In Proc. ACM WIN-ITS, 2007.
[27] Julien Freudiger, Reza Shokri, and Jean-Pierre Hubaux. On the optimal placement of mix zones. In Proc. PETS, 2009.
[28] Balaji Palanisamy and Ling Liu. Mobimix: Protecting location privacy with mix-zones over road networks. In Proc. IEEE ICDE, 2011.
[29] J. Freudiger, M. H. Manshaei, J. P. Hubaux, and D. C. Parkes. Non-cooperative location privacy. In IEEE Transactions on Dependable and Secure Computing, 2014.
[30] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. In Proc. International Conference on Pervasive Services, 2005.
[31] John Krumm. Realistic driving trips for location privacy. In Proc. International Conference on Pervasive Computing, 2009.
[32] Kassem Fawaz and Kang G. Shin. Location privacy protection for connected vehicles and smartphone users. In Proc. IEEE CCNS, 2011.
[33] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li. Achieving k-anonymity in privacy-aware location-based services. In Proc. IEEE INFOCOM, 2014.
[34] Rui Chen, Gergely Acs, and Claude Castelluccia. Differentially private sequential data publication via variable-length n-grams. In Proc. ACM CCS, 2012.
[35] Xi He, Graham Cormode, Ashwin Machanavajjhala, Cecilina M. Procopiuc, and Divesh Srivastava. Dpt: Differentially private trajectory synthesis using hierarchical reference systems. In Proc. VLD Endow, 2015.
[36] Mehmot Emre Gurosy, Ling Liu, Stacey Truex, Lei Yu, and Wenqi Wei. Utility-aware synthesis of differentially private and attack-resilient location traces. In Proc. ACM CCS, 2018.
[37] Fan Bai, Daniel D. Stancil, and Hariharan Krishnan. Toward understanding characteristics of dedicated short range communications (DSRC) from a perspective of vehicular network engineers. In Proc. ACM MOBIWOC, 2010.
[38] GraphHopper directions api route planning for your application. https://github.com/graphhopper.
[39] In-vehicle computers. http://www.lannerinc.com/products/vehicle-computers/.
[40] Seoul to reduce speed limit to 50kmh downtown. http://www.koreatimes.co.kr/www/news/nation/2018/12/281_259700.html.
[41] In-vehicle computers. http://www.lannerinc.com/products/vehicle-computers/.
[42] Thomas Mangel, Olivier Klempp, and Hannes Hartenstein. 5.9 ghz inter-vehicle communication at intersections: a validated non-line-of-sight path-loss and fading model. In EURASIP Journal on Wireless Communications and Networking, 2011.
[43] J. Orto, B. Bustamante, and A. Berry. Down the block and around the corner: the impact of radio propagation on inter-vehicle wireless communication. In Proc. IEEE ICDCS, 2009.
[44] Qiang Liu, Shu Wu, Liang Tian, and Tienniu Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In Proc. AAAI, 2016.
[45] Di Yao, Chao Zhang, Jianhui Huang, and Jingping Bi. Serm: A recurrent model for next location prediction in semantic trajectories. In Proc. ACM CIKM, 2017.
[46] TensorFlow: An open-source software library for machine intelligence. https://www.tensorflow.org/.
[47] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from gps trajectories. In Proc. WWW, 2009.
[48] Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. Understanding mobility based on gps data. In Proc. ACM Ubicomp, 2008.
[49] Yu Zheng, Xing Xie, and Wei-Ying Ma. Geoflite: A collaborative social networking service among user, location and trajectory. In IEEE Data Engineering Bulletin, 2010.
[51] Joseph Meyerowitz and Ronit Roy Choudhury. Hiding stars with fire-works: Location privacy through camouflage. In Proc. ACM Mobicom, 2009.

[52] R. Shokri, G. Theodorakopoulos, J. Le Boudec, and J. Hubaux. Quantify-ing location privacy. In Proc. IEEE Symposium on Security and Privacy, 2011.

[53] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Next place prediction using mobility markov chains. In Proc. ACM MPM, 2012.

[54] OpenStreetMap. https://www.openstreetmap.org.

[55] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In Proc. ACM CCS, 2013.

[56] Pravin Shankar, Vinod Ganapathy, and Liviu Iftode. Privately querying location-based services with sybilquery. In Proc. UbiComp, 2009.

[57] E. Levrina and P. Bickel. The earth mover’s distance is the mallows distance: some insights from statistics. In Proc. IEEE International Conference on Computer Vision, 2001.

[58] Baik Hoh and Marco Gruteser. Preserving privacy in gps traces via uncertainty-aware path cloaking. In Proc. ACM CCS, 2007.

[59] V. Bindschaedler and R. Shokri. Synthesizing plausible privacy-preserving location traces. In Proc. IEEE Symposium on Security and Privacy, 2016.

[60] George Siokkas, Evangelos Skodras, and Evangelos Dermatas. Traffic lights detection in adverse conditions using color, symmetry and spatiotemporal information. In Proc. Computer Vision Theory and Applications, 2012.

[61] Guo Mu, Zhang Xinyu, Li Deyi, Zhang Tianlei, and An Lifeng. Traffic light detection and recognition for autonomous vehicles. In The Journal of China Universities of Posts and Telecommunications, 2015.

[62] Driverless car market watch. http://www.driverless-future.com/.

JAEMIN LIM received the B.S. and M.S. degree in computer engineering from Hanyang University, South Korea, where he is currently pursuing the Ph.D. degree. His research interests are systems security and user privacy.

KIYEON KIM received the B.S. and M.S. degree in computer engineering from Hanyang University, South Korea, where he is currently pursuing the Ph.D. degree. His research interests are computer networks and systems security.

HYUNWOO YU received the B.S. and M.S. degree in computer engineering from Hanyang University, South Korea, where he is currently pursuing the Ph.D. degree. His research interests include computer networks, mobile systems, and network security.

SUK-BOK LEE (M’11) is currently an Associate Professor with the Department of Computer Science and Engineering, Major in Bio Artificial Intelligence, Hanyang University, South Korea. His research interests include computer networks, mobile systems, and wireless networking.