Footprint: Detecting Sybil Attacks in Urban Vehicular Networks

Shan Chang, Yong Qi, *Member*, *IEEE*, Hongzi Zhu, *Member*, *IEEE*,
Jizhong Zhao, *Member*, *IEEE*, and Xuemin (Sherman) Shen, *Fellow*, *IEEE*

Abstract—In urban vehicular networks, where privacy, especially the location privacy of anonymous vehicles is highly concerned, anonymous verification of vehicles is indispensable. Consequently, an attacker who succeeds in forging multiple hostile identifies can easily launch a Sybil attack, gaining a disproportionately large influence. In this paper, we propose a novel Sybil attack detection mechanism, Footprint, using the trajectories of vehicles for identification while still preserving their location privacy. More specifically, when a vehicle approaches a road-side unit (RSU), it actively demands an authorized message from the RSU as the proof of the appearance time at this RSU. We design a *location-hidden* authorized message generation scheme for two objectives: first, RSU signatures on messages are *signer ambiguous* so that the RSU location information is concealed from the resulted authorized message; second, two authorized messages signed by the same RSU within the same given period of time (*temporarily linkable*) are recognizable so that they can be used for identification. With the temporal limitation on the linkability of two authorized messages, authorized messages used for long-term identification are prohibited. With this scheme, vehicles can generate a location-hidden trajectory for location-privacy-preserved identification by collecting a consecutive series of authorized messages. Utilizing social relationship among trajectories according to the similarity definition of two trajectories, Footprint can recognize and therefore dismiss "communities" of Sybil trajectories. Rigorous security analysis and extensive trace-driven simulations demonstrate the efficacy of Footprint.

Index Terms—Sybil attack, location privacy, signer-ambiguous signature, urban vehicular networks, location-hidden trajectory.

1 Introduction

Over the past two decades, vehicular networks have been emerging as a cornerstone of the next-generation Intelligent Transportation Systems (ITSs), contributing to safer and more efficient roads by providing timely information to drivers and concerned authorities. In vehicular networks, moving vehicles are enabled to communicate with each other via intervehicle communications as well as with road-side units (RSUs) in vicinity via roadside-to-vehicle communications. In urban vehicular networks where the privacy, especially the location privacy of vehicles should be guaranteed [1], [2], vehicles need to be verified in an anonymous manner. A wide spectrum of applications in such a network relies on collaboration and information aggregation among participating vehicles. Without identities of participants, such applications are vulnerable to the Sybil attack where a malicious vehicle masquerades as multiple identities [3], overwhelmingly influencing the result. The consequence of Sybil attack happening in vehicular networks can be vital. For example, in safety-related applications such as hazard warning, collision avoidance, and passing assistance, biased results caused by a Sybil attack can lead to severe car accidents. Therefore, it is of great importance to detect Sybil attacks from the very beginning of their happening.

Detecting Sybil attacks in urban vehicular networks, however, is very challenging. First, vehicles are anonymous. There are no chains of trust linking claimed identities to real vehicles. Second, location privacy of vehicles is of great concern. Location information of vehicles can be very confidential. For example, it can be inferred that the driver of a vehicle may be sick from knowing the vehicle is parking at a hospital. It is inhibitive to enforce a one-to-one correspondence between claimed identities to real vehicles by verifying the physical presence of a vehicle at a particular place and time. Third, conversations between vehicles are very short. Due to high mobility of vehicles, a moving vehicle can have only several seconds [4] to communicate with another occasionally encountered vehicle. It is difficult to establish certain trustworthiness among communicating vehicles in such a short time. This makes it easy for a malicious vehicle to generate a hostile identity but very hard for others to validate. Furthermore, short conversations among vehicles call for online Sybil attack detection. The detection scheme fails if a Sybil attack is detected after the attack has terminated.

To eliminate the threat of Sybil attacks, it is straightforward to explicitly bind a distinct authorized identity (e.g.,

• *S. Chang, Y. Qi, and J. Zhao are with the School of Computer Science and Technology, Xi'an Jiaotong University, No.1 West Building, No. 28, Xianning West Road, Xi'an, Shaanxi 710049, P.R. China.*
E-mail: changshan.07@stu.xjtu.edu.cn, {qiy, zjz}@mail.xjtu.edu.cn.
• *H. Zhu is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Room A-303, SEIEE No. 5 Building, 800 Dong Chuan Road, Min Hang, Shanghai 200240, P.R. China.*
E-mail: hongzi@cs.sjtu.edu.cn.
• *X. Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Room-4155, EIT building, 200 University Avenue West, Waterloo, ON N2L 3G1, Canada.*
E-mail: xshen@bbcr.uwaterloo.ca.

Manuscript received 2 Apr. 2011; revised 24 Sept. 2011; accepted 26 Sept. 2011; published online 19 Oct. 2011.
Recommended for acceptance by X. Cheng.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2011-04-0199.
Digital Object Identifier no. 10.1109/TPDS.2011.263.

PKI-based signatures) [5], [6], [8] to each vehicle so that each participating vehicle can represent itself only once during all communications. Using explicit identities of vehicles has the potential to completely avoid Sybil attacks but violates the anonymity concern in urban vehicular networks. As an alternative scheme, resource testing [9], [10], [11] can be conducted to differentiate between malicious and normal vehicles, where the judgment is made whether a number of identities possess fewer resources (e.g., computational and storage ability) than would be expected if they were distinct. This scheme fails in heterogeneous environments where malicious vehicles can easily have more resources than normal ones. Considering the fact that a vehicle can present itself at only one location at a time, localization techniques or other schemes like the Global Positioning System (GPS) aiming to provide location information of vehicles can be exploited to detect hostile identities. However, these schemes often fail in complicated urban settings (e.g., bad GPS signals due to urban canyons, inaccurate localizations due to highly dynamic wireless signal quality). Recently, two group-signature-based schemes [16], [17] have been proposed, where a message received from multiple distinct vehicles is considered to be trustworthy. Using group signatures can provide anonymity of vehicles and suppress Sybil attacks by restraining duplicated signatures signed by the same vehicles. One practical issue of these schemes is that different messages with similar semantics may be ignored from making the decision, which leads to a biased or no final decision. As a result, there is no existing successful solution, to the best of our knowledge, to tackling the online Sybil attack detection problem in urban vehicular networks.

In this paper, we propose a novel Sybil attack detection scheme Footprint, using the trajectories of vehicles for identification while still preserving the anonymity and location privacy of vehicles. Specifically, in Footprint, when a vehicle encounters an RSU, upon request, the RSU issues an authorized message for this vehicle as the proof of its presence at this RSU and time. Intuitively, authorized messages can be utilized to identify vehicles since vehicles located at different areas can get different authorized messages. However, directly using authorized messages will leak location privacy of vehicles because knowing an authorized message of a vehicle signed by a particular RSU is equivalent to knowing the fact that the vehicle has showed up near that RSU at that time. In Footprint, we design a *location-hidden* authorized message generation scheme for two purposes. First, RSU signatures on messages are *signer-ambiguous* which means an RSU is anonymous when signing a message. In this way, the RSU location information is concealed from the final authorized message. Second, authorized messages are *temporarily linkable* which means two authorized messages issued from the same RSU are recognizable if and only if they are issued within the same period of time. Thus, authorized messages can be used for identification of vehicles even without knowing the specific RSUs who signed these messages. With the temporal limitation on the linkability of two authorized messages, authorized messages used for long-term identification are prohibited. Therefore, using authorized messages for identification of vehicles will not harm anonymity of vehicles.

To be uniquely identified, a vehicle collects a consecutive series of authorized messages as it keeps traveling. Such a sequence of authorized messages constitutes a *trajectory* of this vehicle. In Footprint, a vehicle is free to start a new trajectory by using a new temporary public key. Furthermore, a malicious vehicle can abuse this freedom to elaborately generate multiple trajectories, trying to launch a Sybil attack. Based on the observation that Sybil trajectories generated by a malicious vehicle are very alike, Footprint establishes the relationship between a pair of trajectories according to our definition of similarity. With this relationship, Sybil trajectories generated by the same malicious vehicle form a "community." By finding and eliminating "communities" of Sybil trajectories, Footprint can detect and defend against Sybil attacks.

The advantages of Footprint are fourfold. First, Footprint does not need the identities of vehicles, which ensures the anonymity of vehicles. Second, no geographical information is leaked in Footprint, which guarantees the location privacy of vehicles. Third, Footprint only needs each vehicle to be equipped with a cheap commercial GPS receiver and DSRC wireless communication module. Last, Sybil attack detection can be online independently conducted by a conversation holder (e.g., an individual vehicle or an RSU) which initializes a conversation among vehicles. Besides the advantages, the main limitation of Footprint is that Footprint requires an infrastructure of RSUs and a trust authority (TA) existing in the system in order to generate trajectories and establish trust among entities, respectively. We verify that Footprint can achieve all design objectives through security, privacy, and performance analysis and extensive trace-driven simulations which involve 2,100 taxies in Shanghai city. Footprint can largely restrict Sybil attacks and enormously reduces the impact of Sybil attacks in urban settings (above 98 percent detection rate).

The remainder of this paper is organized as follows. Section 2 introduces previous work on the Sybil attack detection problem. In Section 3, we describe the system and attack models in vehicular networks, pointing out the requirements for designing Sybil attack detection schemes in urban vehicular networks. Section 4 elaborates the design of Footprint. In Section 5, we present the security, privacy, and performance analysis of Footprint. Several design issues that may be encountered in practice are discussed in Section 6. In Section 7, we conduct trace-driven simulations to evaluate the performance of Footprint and present the results. Finally, we give concluding remarks and outline the directions for future work in Section 8.

2 Related Work

While it was first described and formalized by Douceur [3], the Sybil attack has been a severe and pervasive problem in many forms. In a Sybil attack, an attacker can launch a Sybil attack by forging multiple identifies, gaining a disproportionately large influence. In the literature, there have been many different approaches proposed to detect or mitigate the attack.

Many studies have followed Douceur's approach, focusing on how to establish trust between participating entities based on trusted public key cryptographies or certificates in distributed systems, for example, P2P systems [3], [5], sensor networks [6], [7] and mobile ad hoc networks [8]. Although deploying trusted certificates is the only approach that has the potential to completely eliminate Sybil attacks, it also violates both anonymity and location privacy of entities. In addition, most of these schemes rely on a centralized authority that must ensure each entity is assigned exactly one identity. Moreover, it is possible for an attacker to violate the assumption, getting more than one identities. This mechanism also has the problem of key revocation which is challenging, particularly in wireless mobile networks.

Another category of Sybil attack detection schemes is based on resource testing [9], [10], [11]. The goal of resource testing is to determine if a number of identities possess fewer resources than would be expected if they were independent. The resources being tested can be computing ability, storage ability, and network bandwidth, as well as IP addresses. These schemes assume that entities have homogeneous hardware configurations. In vehicular networks, this assumption cannot hold since malicious vehicles can easily have more powerful resources than the normal vehicles.

SybilGuard [12] is an interesting scheme studying the social network among entities. In this scheme, human-established real-world trust relationship among users is used for detecting Sybil attacks. Since even the attacker can generate as many as Sybil identities, building relationship between honest users and Sybil identities is much harder. Thus, there exists a small "cut" on the graph of trust relationship between the forged identities and the real ones. However, this scheme cannot be used in vehicular networks, since it is very challenging to establish such trust relationship among vehicles. This is because vehicles are highly mobile. Communications often happen among temporarily met and unfamiliar vehicles.

To exploit the fact that one single vehicle cannot present at multiple locations at the same time, Bouassida et al. [13] have proposed a detection mechanism utilizing localization technique based on Received Signal Strength Indication (RSSI). In this scheme, by successively measuring the RSSI variations, the relative locations among vehicles in vicinity can be estimated. Identities with the same estimated locations are considered as Sybil vehicles. In practice, the complicated outdoor environments can dramatically affect the wireless signal propagation so that RSSI measurements are highly time variant even measured at the same location. Xiao et al. [14] have proposed a Sybil attack detection scheme where the location of a particular vehicle can be determined by the RSSI measurements taken at other participating vehicles. In addition to the inaccuracy of RSSI measurements, this scheme also needs all neighboring vehicles to collaborate which may suffer a Sybil attack against the detection scheme itself. Zhou et al. [15] have proposed a privacy-preserving Sybil attack detection scheme using pseudonyms. In the scheme, the trust authority distributes a number of pseudonyms for each vehicle. Abused pseudonyms can be detected by RSUs. Since RSUs are heavily involved in the detection process, this scheme requires the full coverage of RSUs in the field. It is infeasible in practice due to the prohibitive cost. Furthermore, in such a scheme, vehicles should managed by a centralized trusted center. Each time RSU detects suspicious pseudonyms, it should send all the pseudonyms to the trust center for further decision, which makes the trust center be the bottleneck of the detection.

Recently, two group-signature-based schemes [16], [17] have been proposed, ensuring that a verifier vehicle can identify those trustworthy messages from messages sent from neighboring vehicles. A message sent from a neighboring vehicle is said to be trustworthy if the content of the message is identical with at least a certain number of messages sent from other neighboring vehicles. To suppress duplicated messages from the same vehicle, particular group signature schemes are adopted for vehicles to sign on messages so that the anonymity of each vehicle can be achieved. Meanwhile, if a vehicle generates two signatures on the same message, these two signatures can be recognized by the verifier vehicle. One practical issue of these schemes is that they cannot handle similar but different messages. It is often the case that multiple vehicles observing the same driving environment will generate different messages with very similar semantics. In this case, the resolved trustworthy messages might be a minority of all observations which results in a biased or no final decision.

The most relevant work to Footprint is the Sybil attack detection schemes proposed in [18], [19]. In these schemes, a number of location information reports about a vehicle are required for identification. In [18], an RSU periodically broadcasts an authorized time stamp to vehicles in its vicinity as the proof of appearance at this location. Vehicles collect these authorized time stamps which can be used for future identity verification. In [19], trajectories made up of consecutive time stamps and the corresponding public keys of RSUs are used for identification. However, these schemes did not take location privacy into consideration since RSUs use long-term identities to generate signatures. As a result, the location information of a vehicle can be inferred from the RSU signatures it collects. In Footprint, authorized messages issued from RSUs are signer-ambiguous which means the information about the location where the authorized message was issued is concealed, and temporarily linkable which means using a single trajectory for long-term identification of a vehicle is prohibited. Therefore, the privacy of location information and identity of vehicles are preserved in Footprint.

3 Models and Design Goals

3.1 System Model and Assumptions

In vehicular networks, a moving vehicle can communicate with other neighboring vehicles or RSUs via intervehicle communications and roadside-to-vehicle communications. Fig. 1 illustrates the architecture of the system model, which consists of three interactive components:

- **RSUs:** can be deployed at intersections or any area of interest (e.g., bus stations and parking lot entrances). A typical RSU also functions as a wireless AP (e.g., IEEE 802.11x) which provides wireless access to users within its coverage. RSUs are interconnected (e.g., by a dedicated network or through the Internet via cheap ADSL connections) forming a RSU backbone network.

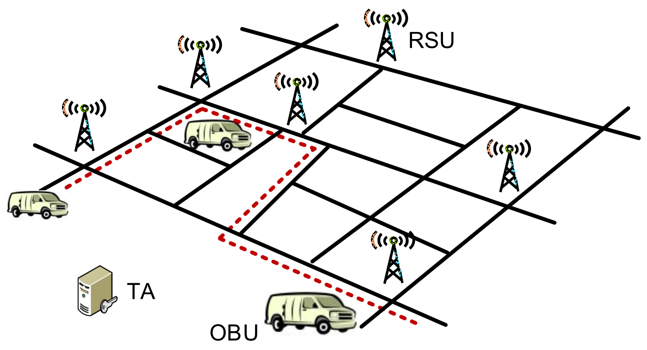

Fig. 1. An illustration of the system model, where the dash line indicates the travel route of a vehicle. As the vehicle traverses the area, it will encounter multiple RSUs, typically deployed at intersections.

- **On-board units (OBUs):** are installed on vehicles. A typical OBU can equip with a cheap GPS receiver and a short-range wireless communication module (e.g., DSRC IEEE 802.11p [20]). A vehicle equipped with an OBU can communicate with an RSU or with other vehicles in vicinity via wireless connections. For simplicity, we simply refer to a vehicle as a vehicle equipped with an OBU in the rest of this paper. A vehicle can be malicious if it is an attacker or compromised by an attacker.
- **Trust authority:** is responsible for the system initialization and RSU management. The TA is also connected to the RSU backbone network. Note that the TA does not serve vehicles for any certification purpose in Footprint. A vehicle can claim as many arbitrary identities as it needs.

In this work, we make the following assumptions:

Assumption 1. *The TA and all RSUs are fully trustworthy.*

Assumption 2. *The RSUs are synchronized.*

Synchronization among RSUs is easy to achieve since all RSUs are interconnected by the RSU backbone network.

Assumption 3. *The mobility of vehicles is independent.*

This means individual vehicles should move independently and therefore would not travel along the same route for all the time.

3.2 Attack Model

In order to launch a Sybil attack, a malicious vehicle must try to present multiple distinct identities. This can be achieved by either generating legal identities or by impersonating other normal vehicles. With the following capabilities, an attacker may succeed to launch a Sybil attack in vehicular networks:

- *Heterogeneous configuration:* malicious vehicles can have more communication and computation resources than honest vehicles. For example, a malicious vehicle can mount multiple wireless cards, physically representing different communication entities. Furthermore, having more powerful resources can also fail those resource testing schemes for detecting Sybil attacks.
- *Message manipulation:* due to the nature of open wireless channels, the attacker can eavesdrop on nearby communications of other parties. Thus, it is possible that the attacker gets and interpolates critical information needed to impersonate others.

In any decision-making procedure based on reports sent from a number of individual vehicles, if an attacker succeeds in presenting multiple independent identities, it can launch Sybil attacks against honest vehicles where the attacker can inject multiple false reports via multiple identities into the final decision. Upon Sybil attacks happening, the final results may be biased due to the influence of false reports sent from attackers.

3.3 Design Goals

The design of a Sybil attack detection scheme in urban vehicular networks should achieve three goals:

1. *Location privacy preservation:* a particular vehicle would not like to expose its location information to other vehicles and RSUs as well since such information can be confidential. The detection scheme should prevent the location information of vehicles from being leaked.
2. *Online detection:* when a Sybil attack is launched, the detection scheme should react before the attack has terminated. Otherwise, the attacker could already achieve its purpose.
3. *Independent detection:* the essence of Sybil attack happening is that the decision is made based on group negotiations. To eliminate the possibility that a Sybil attack is launched against the detection itself, the detection should be conducted independently by the verifier without collaboration with others.

4 System Design

4.1 Overview

In general, Footprint integrates three elegant techniques namely, infrastructure construction, location-hidden trajectory generation, and Sybil attack detection.

More specifically, we adopt an incremental methodology to deploy RSUs. In the end, a limited number of available RSUs can achieve the maximum service coverage in terms of served traffic amount as well as good fairness in terms of geographical distribution. After the deployment of RSUs, a vehicle can require authorized messages from each RSU it passes by as a proof of its presence there. We adopt an *event-oriented linkable ring signature* scheme [24] for RSUs to issue authorized messages for vehicles. Such authorized messages are *location hidden* which refers to that RSU signatures are signer ambiguous and the authorized messages are temporarily linkable. Furthermore, a set of consecutive authorized messages issued for a vehicle are tightly chained together to form a location-hidden trajectory of the vehicle, which will be utilized for identifying this vehicle in future conversations. During a conversation which is initialized by a vehicle or an RSU, called a *conversation holder*, a participating vehicle should provide its trajectory for verification. With the trajectories sent from all participating vehicles, the conversation holder can conduct online Sybil attack detection according to the similarity relationship between each pair of trajectories. Among all trajectories, Sybil trajectories forged from the same attacker are bound to gather within the same "community." By treating each "community" as one single vehicle, Sybil trajectories can be largely eliminated.

In the following sections, we describe each technique in details.

4.2 Infrastructure Construction

4.2.1 RSU Deployment

In Footprint, vehicles require authorized messages issued from RSUs to form trajectories, which should be statically installed as the infrastructure. When considering the deployment of RSUs, two practical questions are essential, i.e., *where to install RSUs in the city* and *how many of them are sufficient?*

A simple solution is to deploy RSUs at all intersections. This can result fine trajectories with a sufficient number of authorized messages which will facilitate the recognition of a vehicle. However, deploying such a huge number of RSUs in one time is prohibitive due to the high cost.

In contrast, we take an incremental deployment strategy in Footprint, considering the tradeoff between minimizing the number of RSUs and maximizing the coverage of traffic. Specifically, in the early developing stage with a limited number of RSUs, an intersection is chosen if it satisfies two requirements: first, it is geographically at least certain distance far away from all other RSU-equipped intersections; second, it has the maximum traffic volume among all rest intersections without RSUs. The reason for requiring two RSUs at least certain distance far away is to avoid uneven deployment where RSUs are consecutively deployed along a high-traffic-volume road. As more RSUs are available to install, a smaller distance can be used to deploy RSUs according to the above strategy. Given an RSU deployment, two RSUs are said to be neighbors if there exists a path in the underlying road networks along which no other RSUs are installed (see Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.263, for an RSU deployment example).

4.2.2 System Initialization

After completing RSU deployment, in order to function properly, the system first needs to be initialized. The initialization process includes three steps:

1. Setting up TA: the TA first chooses a set of public parameters required for the ring signature scheme which is used for RSUs to sign messages and establishes a pair of public/private key pair $(K_{\mathrm{TA}}^{pub}, K_{\mathrm{TA}}^{pri})$ as well. The public key of the TA K_{TA}^{pub} can be obtained by all RSUs and vehicles in the system through a secure channel. It is used to verify whether a message is authorized by the TA (see Appendix C, available in the online supplemental material, for the detailed initialization process).

2. Setting up RSUs: when a new RSU R_k is added to the system, the TA issues a pair of public/private key pair $(K_{R_k}^{pub}, K_{R_k}^{pri})$ for R_k and sends the public parameters to R_k as well. After all RSUs are registered in the system, the Public Key List (PKL) of all RSUs is broadcasted to all RSUs from the TA via the RSU backbone network. In addition, the IP addresses of its neighboring RSUs of R_k are also notified to R_k. Note that all messages sent from the TA are authorized by the TA using its private key K_{TA}^{pri}.

Complying with the incremental deployment of RSUs, version control is taken by the TA in managing the PKL. More specifically, when new RSUs enroll in the system, the TA updates the PKL and increases its version number. Then, the newest PKL can be broadcasted to all RSUs in the system via the RSU backbone network.

3. Setting up vehicles: For a vehicle to join in the system, it only needs to get the PKL of all RSUs and the public parameters. It can get such information when encountering any RSU or a vehicle with the information. After that, it can construct its own trajectories in the system.

For a vehicle to be up-to-date with the latest PKL, each time it communicates with an RSU, the vehicle can include the version number of PKL it has. The RSU can verify whether the vehicle has the latest version. If not, the RSU will help the vehicle update the PKL.

4.3 Generating Location-Hidden Trajectory

4.3.1 Location-Hidden Authorized Message Generation

In order to be location hidden, authorized messages issued for vehicles from an RSU should possess two properties, i.e., *signer ambiguous* and *temporarily linkable*. The signer-ambiguous property means the RSU should not use a dedicated identity to sign messages. The temporarily linkable property requires two authorized messages are recognizable if and only if they are generated by the same RSU within the same given period of time. Otherwise, a long-term linkability of authorized messages used for identification eventually has the same effect as using a dedicated identity for vehicles.

In this paper, we demonstrate one possible implementation of a location-hidden authorized message generation scheme using linkable ring signature [21]. Linkable ring signature is signer-ambiguous and signatures are linkable (i.e., two signatures can be linked if and only if they are issued by the same signer) as well. Particularly, we choose the linkable ring signature scheme introduced by Dodis et al. [22] and Tsang and Wei [23] for two reasons: first, it has been proved to be secure; second, it has constant signature size. To meet the requirement of temporarily linkable property, we extend the scheme to support the *event-orient linkability* property [24] which guarantees that any two signatures are linkable if and only if they are signed based on the same event by the same RSU.

In our signature scheme, we define an *event* as a period of time within which two signatures issued from the same RSU are linkable. Thus, an RSU signature consists of three parts: *proof of knowledge (pok)*, *event id*, and *link tag*. The *pok* is a proof that the signature on the message $\mathcal{M}$ is legitimate. The *event id* is a fixed-size bit string derived by a secure cryptographic hash function on an event (i.e., a period of time). The *link tag* is generated based on the *event id* and the private key of an RSU. When an event expires, all RSUs in the system simultaneously compute a new *event id* and *link tag* for the next event (next period of time). With time variant link tags, the RSU signatures can meet the temporarily linkable requirement.

An intuitive way to generate authorized messages for vehicles is that an RSU periodically broadcasts authorized time stamps to the vehicles in its vicinity. This method is simple but not secure. Since a time stamp is not specially generated for a particular vehicle, any other vehicle getting such a time stamp by eavesdropping on the wireless channels can claim its presence at this RSU even though it has never been there at that time. Therefore, time stamps should be generated for individual vehicles. In Footprint, when a vehicle v_i approaches an RSU R_k, it demands a time stamp from R_k, using a jth temporarily generated key pair $(K^{pub}_{v_i,j}, K^{pri}_{v_i,j})$ (v_i can generate a set of temporary key pairs in advance). Upon request, R_k generates a message $\mathcal{M}$ for v_i, which includes $K^{pub}_{v_i,j}$, and a time stamp indicating the time when this message is generated. Then, R_k signs on the message $\mathcal{M}$ and sends $\mathcal{M}$ together with the signature, denoted as $\mathcal{M} \parallel S_{R_k}(\mathcal{M})$, back to v_i (see Appendix D, available in the online supplemental material, for the details of signature generation).

4.3.2 Message Verification

As the proof that a vehicle v_i was present near certain RSU R_k at certain time, an authorized message issued for v_i can be verified by any entity (e.g., a vehicle or an RSU) in the system.

In the case that an entity needs to verify v_i, v_i will sign on an authorized message $\mathcal{M} \parallel S_{R_k}(\mathcal{M})$ generated by RSU R_k using $K^{pri}_{v_i,j}$ and then send

$$L_{R_k} = \mathcal{M} \parallel S_{R_k}(\mathcal{M}) \parallel S_{K^{pri}_{vi,j}}(\mathcal{M} \parallel S_{R_k}(\mathcal{M}))$$

to the entity. The message verification process consists of two steps:

1. Ownership verification: The entity first takes $K^{pub}_{v_i,j}$ from $\mathcal{M}$, checking whether $V_{K^{pub}_{v_i,j}}(S_{K^{pri}_{v_i,j}}(\mathcal{M}, S_{R_k}(\mathcal{M}))) = \mathcal{M} \parallel S_{R_k}(\mathcal{M})$. Since any $K^{pri}_{v_m,x}$ other than $K^{pri}_{v_i,j}$ will fail the test, an authorized message cannot be misused by other vehicles because only v_i knows $K^{pri}_{v_i,j}$ which is pairwise with $K^{pub}_{v_i,j}$ contained in $\mathcal{M}$. Therefore, if the test stands, it means $\mathcal{M}$ is exclusively generated for v_i rather than for other vehicles.

2. Legitimacy verification: If the authorized message passes the ownership verification, the entity further examines whether the signature contained in the authorized message is signed by a legitimate RSU in the system (see Appendix E, available in the online supplemental material, for the details of signature verification).

In the case that v_i fails in either step, the entity will consider v_i as a malicious vehicle and ignore any further actions of v_i.

4.3.3 Trajectory-Encoded Message

Intuitively, an authorized message issued from an RSU can be used to identify a vehicle. However, it is often the case that two or more authorized messages may have the same link tag. In this case, it is hard to tell whether these messages belong to different vehicles.

With the independent mobility assumption, as two vehicles move along, the probability for the pair of vehicles having exactly the same trajectories is slim. Therefore, it is feasible to use trajectories to exclusively represent corresponding vehicles as long as those trajectories are sufficiently long. With authorized messages, a straightforward method for a vehicle to present its trajectory is to sort all its authorized messages into a sequence according to time. Thus, in future conversations, the vehicle can use this sequence of authorized messages to identify itself. This method is simple but inefficient because each time when the vehicle needs to be identified in a conversation, all messages in the sequence should be sent to the conversation holder for verification. This will cost tremendous wireless bandwidth and computational resources. Furthermore, a malicious vehicle can easily forge a huge number of fake trajectories by arbitrarily picking a subset of authorized messages as long as these messages are in the right order of time. Since authorized messages are location hidden, the conversation holder cannot tell whether a provided trajectory is an actual one or a forged one.

In Footprint, we embed the trajectory of a vehicle into an authorized message. Specifically, upon the starting of a new event, besides computing the new event id and link tag for the new event, an RSU also informs all its neighboring RSUs with the new generated link tag. During the new event, when a vehicle first meets an RSU R_k, it requests an authorized message $\mathcal{M} \parallel S_{R_k}(\mathcal{M})$ from R_k using temporary key pair $(K^{pub}_{v_i,j}, K^{pri}_{v_i,j})$ following the procedure as described in Section 4.3.1. As this vehicle moves on and encounters another RSU R_l, it first chooses a new temporary key pair $(K^{pub}_{v_i,j+1}, K^{pri}_{v_i,j+1})$. Then, it signs on $K^{pub}_{v_i,j+1} \parallel \mathcal{M} \parallel S_{R_k}(\mathcal{M})$, using $K^{pri}_{v_i+j}$. After that, it requests R_l for a new trajectory-embedded authorized message by sending $K^{pub}_{v_i,j+1} \parallel \mathcal{M} \parallel S_{R_k}(\mathcal{M}) \parallel SK^{pub}_{v_i,j+1}, \mathcal{M}, S_{R_k}(\mathcal{M}))$ to R_l. Upon request, R_l first verifies the authorized message following the procedure described in the above section. If the verification succeeds, R_l further checks whether the link tag in $S_{R_k}(\mathcal{M})$ belongs to one of its neighbors. If yes, R_l constitutes a new message with the new temporary public key of the vehicle $K^{pub}_{v_i,j+1}$, current time stamp, all (link tag, time stamp) pairs contained in $\mathcal{M}$ if any. Then, R_l signs on the new message and sends the trajectory-embedded authorized message back to the vehicle. If the link tag contained in $S_{R_k}(\mathcal{M})$ does not belong to any neighboring RSU of R_l, R_l will treat itself as the first RSU in the trajectory of the vehicle and sign accordingly. This procedure repeats as the vehicle moves. As a result, a trajectory is embedded within a single authorized message (see Appendix F, available in the online supplemental material, for an example of trajectory generation).

Within an event, a vehicle can actively choose to terminate the current trajectory and start a new trajectory at any time by sending only a new temporary public key to an RSU. When an event expires, all RSUs will simultaneously change their link tags. In this case, the system forces all vehicles to start new trajectories.

With trajectory-encoded messages, each time when a vehicle needs to be identified, the vehicle only needs to send a single authorized message to a verifier which extremely reduces the number of verifications from $O(l)$ to $O(1)$, l is the length of the trajectory (i.e., the number of involved RSUs). Moreover, the trajectory encoded in an authorized message is verified and constructed by neighboring RSUs, which largely limits the ability of a malicious vehicle to arbitrarily forge fake trajectories.

4.4 Sybil Attack Detection

During a conversation, upon request from the conversation holder, all participating vehicles provide their trajectory-embedded authorized messages issued within specified

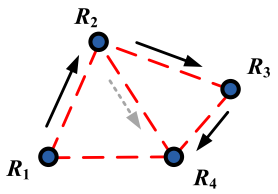

Fig. 2. RSU neighboring relationship and the freedom of trajectory generation can facilitate Sybil trajectory generation. In the above figure, neighboring RSUs (denoted by dots) are connected with dash line. The solid arrows indicate the actual sequence of RSUs a malicious meet and the dash arrow presents a possible forged trajectory.

event for identification. With submitted messages, the conversation holder verifies each trajectory and refuses those vehicles that fail the message verification. After that, the conversation holder conducts online Sybil attack detection before further proceeding with the conversation.

4.4.1 Problem Definition

Recall that, in Footprint, vehicles have wide freedom to create their trajectories. For example, a vehicle is allowed to request multiple authorized messages from an RSU using different temporary key pairs. Thus, a vehicle can use different authorized messages for different conversations. This capability, however, can be leveraged by a malicious vehicle that tries to launch a Sybil attack by using multiple different messages in a single conversation. We define the Sybil attack detection problem as: *Given a set of trajectory-embedded authorized messages within an event, how can the conversation holder recognize real vehicles and Sybil ones?*

The online Sybil attack problem is hard due to three following factors:

First, authorized messages generated for different vehicles are asynchronous. The rationale of using trajectories to represent vehicles is based on the fact that a vehicle cannot present itself at different locations at the same time. The asynchrony of messages makes the judgment directly based on this fact impractical.

Second, authorized messages are temporarily linkable, which means there is no invariable mapping between an RSU signature and the real RSU who signed this signature. Thus, no distance information is available between two RSUs enclosed in any two signatures. This makes the problem even harder since one cannot utilize the time difference between two authorized messages and the distance between the pair of corresponding RSUs to infer whether two messages belong to two distinct vehicles.

Last, a malicious vehicle can abuse the freedom of trajectory generation and the neighbor relationship among RSUs to generate elaborately designed trajectories. For example, in Fig. 2, an attacker can legally generate multiple trajectories which appear different from each other even under a very simple RSU topology. Assume the real path of the attacker is $\{R_1, R_2, R_3, R_4\}$ (indicated by solid arrows). It can start a new trajectory at any RSU by using a different temporary key pair. Therefore, besides the trajectory $\{R_1, R_2, R_3, R_4\}$, trajectories like $\{R_1, R_2, R_3\}$, $\{R_2, R_3, R_4\}$, $\{R_1, R_2\}$, $\{R_2, R_3\}$, $\{R_3, R_4\}$, $\{R_1\}$, $\{R_2\}$, $\{R_3\}$ and $\{R_4\}$ are all legitimate. In addition, knowing the neighboring relationship of R_2 and R_4, the attacker can generate forged trajectories like $\{R_1, R_2, R_4\}$, $\{R_1, R_4\}$, and $\{R_2, R_4\}$ (indicated by the dash arrow). Note that the attacker cannot generate a trajectory like $\{R_1, R_3\}$ because R_1 is not a neighbor of R_3. In the case of this example, R_3 only expects signatures signed by R_2 and R_4.

In the following sections, we present the social relationship between two trajectories according to our definition of similarity. Then, we introduce how to find and remove Sybil trajectories.

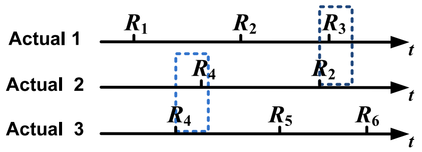

Fig. 3. Checking for distinct trajectories by using a check window (denoted as the box of dotted line) and counting the total number of different RSUs contained in a pair of trajectories.

4.4.2 Social Relationship among Trajectories

We first thoroughly examine the characteristics of forged and actual trajectories.

Features of forged trajectories. Although a malicious vehicle can submit multiple forged trajectories to a conversation holder, these trajectories satisfy two facts. First, a forged trajectory is a proper subset of the actual trajectory. For example, in Fig. 2, all forged trajectories are fully contained in the actual trajectory. Second, any two forged trajectories cannot have two distinct RSUs at the same time. It is true because otherwise the malicious vehicle would appear at two locations at the same time.

Features of actual trajectories. Despite the asynchrony and temporarily linkable properties of authorized messages, there are two basic facts that can be exploited to judge whether two trajectories are from two actual vehicles. First, it is very hard, if not impossible, for a single vehicle to traverse between a pair of RSUs shorter than a time limit. We define such a time limit as *traverse time limit*. Second, within a limited time period, the total number of RSUs traversed by a single vehicle is less than a limit. We define such a limit as *trajectory length limit*. Given a specific RSU deployment, the traverse time limit can be established as the shortest time for a vehicle to travel between any pair of RSUs in the system. The trajectory length limit can be determined as the maximum number of RSUs involved in a trajectory within an event (a trajectory is forced to terminate at the end of an event). Both can be measured based on the distance and speed limitations of each road segment and the layout of RSU deployment.

Based on these features, we first conduct an *exclusion test*, examining whether two trajectories are distinct. There are two cases where a pair of trajectories can pass the test (*positive test*). In the first case, there are two distinct RSUs appearing within a sliding time window (called *check window*) when checking two trajectories. We can set the size of the check window equal to the traverse time limit. For example, in Fig. 3, trajectories $\mathcal{T}_1$ and $\mathcal{T}_2$ are distinct since there exists a pair of different RSUs within the check window (denoted by the box of dash line), i.e., R_2 and R_3. In the second case, the number of RSUs contained in the merged RSU sequence of two trajectories is larger than the trajectory length limit. We merge a pair of trajectories into one RSU sequence by sorting all RSUs contained in the pair of trajectories according to time. In particular, consecutive identical RSUs in the sequence are counted only once. For example, the merged RSU sequence of $\mathcal{T}_1$ and $\mathcal{T}_2$ in Fig. 3 is $\{R_1, R_4, R_2, R_5, R_3, R_6\}$. If the trajectory length limit is 5, then $\mathcal{T}_1$ and $\mathcal{T}_2$ are distinguishable since the length of the sequence is 6. In all other cases, the pair of trajectories fails in the test (*negative test*). For trajectories which are negative to the test, they are treated as suspicious or supplied with insufficient information. For example in Fig. 3, $\mathcal{T}_2$ and $\mathcal{T}_2$ cannot prove that they are mutually exclusive via the exclusion test.

According to the result to the exclusion test, we refine similarity of a trajectory pair T_1 and T_2 as follows:

$$(T_1, T_2) = \begin{cases} -1 & positive\ test \\ \dfrac{|T_1 \cap T_2|}{Min\{|T_1|, |T_2|\}} & negative\ test, \end{cases}$$

where minus one represents that $\mathcal{T}_1$ and $\mathcal{T}_2$ are distinct, $\mathcal{T}_1 \cap \mathcal{T}_2$ denotes the set of common RSUs found when checking $\mathcal{T}_1$ and $\mathcal{T}_2$ using the check window and $|\cdot|$ represents the size of a set or the length of a trajectory. For example, the similarity of $\mathcal{T}_1$ and $\mathcal{T}_3$ in Fig. 3 is minus one and that of $\mathcal{T}_2$ and $\mathcal{T}_3$ is 0.5.

The reason that we define above similarity of a pair of trajectories is twofold. First, any two forged trajectories issued from a malicious vehicle are similar (i.e., a non-negative similarity value). That is to say that all forged trajectories from a malicious vehicle form a social "community" within which any two members have a connection. Second, a trajectory provided by an honest vehicle may have connections with several other trajectories but the probability for this trajectory to form a large "community" is small. This is because an actual trajectory is supposed to contain a sufficient number of RSUs which increase the probability to exclusively differ from others via the exclusion test.

4.4.3 Eliminating Sybil Communities

With the observation in the above section, the Sybil attack detection problem can be well solved by finding an efficient algorithm to eliminate all possible "communities" of Sybil trajectories. However, the problem of finding all Sybil "communities" within a given set of trajectories is very hard. We have the following Theorem.

Theorem 1. *With the definition of relationship between two trajectories in terms of similarity, the problem of finding all "communities" of Sybil trajectories within a given set of trajectories is NP-complete.*

Proof. Assume each trajectory in the set is a vertex in an undirected graph. We define an edge between two vertices if the corresponding trajectories have a non-negative similarity value. To find all "communities" in the trajectory set is equal to find all complete subgraphs (called cliques). Finding all cliques in a graph is a well-known NP-complete problem. This completes the proof. □

In Footprint, we take an iterative procedure to get all Sybil "communities." Specifically, we first generate a corresponding graph according to the procedure described in the Theorem proof. Then, iteratively, we pick a maximum clique each time in the graph and delete all vertices in the clique and all corresponding edges from the graph until there are no more vertices left in the graph. The reason that we pick the maximum clique each time is twofold: First, in order to launch a Sybil attack, a malicious vehicle expects to achieve multiple identifications, which requires the attacker to issue a sufficient number of forged trajectories. This will form a big-sized clique in contrast to those cliques made up of honest vehicles; Second, because the order in which we remove cliques from the graph does not change an original clique from being a clique in the left graph, removing the max clique each time helps shrink the size of the graph, which improves the Sybil attack detection performance. If there are multiple maximum cliques found, we choose the one with the largest sum of similarity associated with all edges. The reason is that a larger similarity means two trajectories are more alike, which are more likely from a malicious vehicle. With each picked maximum clique, we choose the trajectory with the longest length as a legal trajectory and discard all other trajectories in the clique. In this way, a malicious vehicle is allowed to represent itself once no matter how many forged trajectories it has generated.

To pick a maximum clique from the graph, we adopt a heuristic branch-and-bound algorithm [25]. Each time the vertices are first sorted by a greedy vertex coloring algorithm. Then, the search starts from the first vertex. Considering the vehicular application scenario where conversations among vehicles are supposed to be short, we set up a timer for searching the maximum clique. When the timer expires, the currently found clique is returned.

5 Analysis

5.1 Security Analysis

As described in Section 3, a malicious vehicle can easily obtain messages between two other communicating entities by eavesdropping on the wireless channels. In Footprint, all messages are delivered via wireless communication. If a malicious vehicle can succeed in using authorized messages issued for other vehicles, it can masquerade as multiple identities, launching a Sybil attack. The Footprint design is secure in terms of defending:

1. *Against the message replay attacks:* In Footprint, any attempt to misuse authorized messages overheard from other vehicles fails. This is because an authorized message needs first to be verified before it can be used for identifying a vehicle. In the message verification procedure, in order to pass the ownership verification, the attacker must know the temporary private key of the original owner of this message, which is impossible to achieve.

2. *Against the integrity attack:* In Footprint, the attacker cannot interpolate the content of a trajectory either. This is because the integrity of a trajectory is guaranteed by the signature of neighboring RSUs which are fully trustworthy. Verifiers conduct legitimacy verification described in Section 4.3.2 to examine whether the signatures of trajectories are signed by legitimate RSUs in the system. Hence, any trajectory without being authorized by legitimate RSUs will be rejected.

We now analyze the secure level of Footprint with regard to defending against Sybil attacks.

In Footprint, a malicious vehicle can collect as many trajectories as it needs. Upon trying to launch a Sybil attack, the malicious vehicle can also submit as many Sybil trajectories as it needs to a conversation. In addition, the malicious vehicle can also overhear authorized messages sent from other participating vehicles. As a result, the malicious vehicle knows all other trajectories provided to the conversation holder.

Given the Sybil attack detection mechanism, for a Sybil trajectory $\mathcal{T}_1$ to successfully present a Sybil identity, $\mathcal{T}_1$ should be longer than the length of those actual trajectories that are similar with $\mathcal{T}_1$. On the other hand, in order to gain a disproportionately large influence in the conversation, the malicious vehicle should manage to attain enough number of Sybil identities. These two conditions, however, are contradictory because as the length of Sybil trajectories increases, the number of Sybil trajectories decreases very fast. Moreover, since honest vehicles tend to provide their full trajectories so as to be exclusively distinguished, the malicious vehicle has to provide longer Sybil trajectories in order to outstand from possible "communities" of similar trajectories. It is possible only when the malicious vehicle can provide a set of not-so-similar Sybil trajectories which are comparable with actual trajectories provided by honest vehicles in terms of quantity and the trajectory length. This requires the malicious vehicle has much higher mobility than other vehicles, which is not feasible due to the urban settings (e.g., traffic control, speed limitations, traffic condition). In summary, although it cannot fully eliminate the threat of Sybil attacks, Footprint can largely restrict Sybil attacks from happening and enormously reduce the impact even if a Sybil attack happens.

We will extensively evaluate the performance of Footprint in distinguishing honest vehicles from malicious ones via trace-driven simulations in Section 7.

5.2 Privacy Analysis

In addition to security concerns, Footprint can meet the requirement for location privacy preservation of vehicles.

Specifically, a trajectory-embedded authorized message has signer-ambiguous and temporarily linkable properties. With the signer-ambiguous property, the RSU signature contained in the message is anonymous which makes an attacker unable to determine which RSU actually signed the message. Thus, no location information can be inferred by knowing a RSU signature. With the temporarily linkable property, RSUs change their link tags on every new event which means remembering a previous link tag of a RSU does not help an attacker identify this RSU in any other event. Therefore, even if an attacker conducts a field testing by recording the locations of RSUs and their corresponding link tags, it can only log a small number of RSUs for a short period of time.

In addition, as conversation holders randomly choose a previous event and request all participating vehicles to provide a trajectory during this event, a vehicle uses different trajectories generated in different events for identification. Thus, because of the temporarily linkable property of authorized messages, an attacker cannot infer the connection between two trajectories generated in different events. Thus, an attacker cannot track a vehicle.

5.3 Performance Analysis

We analyze the performance of Footprint in terms of computational complexity of the signature generation and verification algorithms and the Sybil attack detection algorithm.

In the signature generation and verification schemes, there are four kinds of operations, i.e., modular addition, modular multiplication, modular exponentiation, and secure cryptographic hash, denoted as *Add*, *Mul*, *Exp*, and *Hash*, respectively. Since the *Exp* and *Hash* operations are far more computationally expensive than the other two operations, we use the number of *Exp* and *Hash* operations to analyze the computational complexity of these two schemes.

In generating or verifying a signature, most of the operations can be conducted in advance. For signing a message, an RSU only needs to compute a hash value (other cheap operations are ignored) for online signing a message. In the case of verifying a signature, a verifier (e.g., a vehicle or an RSU) only needs to conduct 27 *Exp* and one *Hash* operations (see Appendix G, available in the online supplemental material, for the detailed analysis).

In the Sybil attack detection algorithm, a conversation holder who conducts the detection will first verify all provided trajectory-embedded messages. Then, it compares each pair of legal trajectories and uses the heuristic branch-and-bound algorithm to remove Sybil communities. Given n trajectories, the complexity of pairwise trajectory comparison is $O(n^2)$ and the worst case running time complexity of removing Sybil communities is $O(3^{n/2})$.

6 Design Issues

This section discusses some design issues that Footprint may encounter in practice.

Multiple authorized messages. When a vehicle requests an authorized message from an RSU, it is possible that there are multiple RSUs receiving the request and signing a message simultaneously for this vehicle (e.g., in a dense deployment). As a result, the vehicle may get multiple messages signed from different RSUs (i.e., the vehicle can get multiple legitimate trajectories) which can be leveraged by a malicious vehicle to launch Sybil attacks. One simple solution is that while deploying RSUs, the transmission power of RSUs can be properly configured so that there is no coverage overlap between two neighboring RSUs. Thus, a vehicle can only communicate with at most one RSU at one time. More sophisticated methods may need collaboration among neighboring RSUs. For example, a small set of neighboring RSUs can coordinate to localize a vehicle based on their RSSI measurements and select a proper RSU to communicate with the vehicle.

Scalability in terms of the number of verification. Due to the high mobility of vehicles, the duration of interactions between RSUs and vehicles and between vehicles are very short. This may arouse the scalability concern, i.e., how many vehicles a particular RSU or a vehicle is able to interact in a short period of time like seconds. If the generation or verification of signatures is not very efficient, it is possible that a vehicle fails to obtain an authorized message from an RSU before it runs out of the communication range of the RSU. In Footprint, for trajectory verification, only one signature should be verified, which contains a computation of 27 modular exponentiations. With a 512-bit security parameter, it takes roughly 52.38 ms (i.e., 19 vehicles/second) for a 3 GHz processor to complete the whole verification. The average contact time for two vehicles in urban settings is about 10 seconds [4] which is sufficiently long for verifying hundreds of signatures. Therefore, our signature scheme is practical in urban vehicular scenario.

Increasing length of messages. In Footprint, the current trajectory information is required both for an RSU to sign a new message and for other vehicles to verify. As the trajectory continues to grow, the message size is also linearly increasing which consumes more communication bandwidth. Furthermore, in vehicular scenario, where the wireless link quality is very dynamic, long messages may suffer failures. In our scheme, a trajectory is composed of consecutive pairs of RSU link tags and time stamps. The typical length of a message is $68{\cdot}n$ bytes, where n is the length of the trajectory contained in the message. In Footprint, in order to satisfy the temporary linkable property, a short period of time should be chosen as an event. In this case, the n is relatively small (e.g., several tens). This also helps to limit the size of a message.

False alarm of Sybil trajectories. In the Sybil attack detection scheme, it is possible that a trajectory of an honest vehicle could be mixed with other trajectories (either malicious Sybil trajectories or other honest ones) especially when the length of the trajectory is short. This will cause false alarm of Sybil trajectories. This issue can be largely mitigated by comparing multiple sets of trajectories issued in different events. If the probability for an honest trajectory in an event of a vehicle being treated as Sybil is p, then the probability that the vehicle being successfully identified is $(1-p)^m$ when using trajectories in m events. For example, when $p = 0.2$ and $m = 3$, the success probability is above 99 percent.

7 Performance Evaluation

7.1 Methodology

In this section, we examine the performance of Footprint in recognizing forged trajectories (issued by malicious vehicles) and actual ones (provided by honest vehicles) through trace-driven simulations. We consider two key metrics:

1. **False positive error:** is the proportion of all actual trajectories that are incorrectly identified as forged trajectories.
2. **False negative error:** is the proportion of all forged trajectories that are falsely identified as actual trajectories.

In the following simulations, we use the digital map of Yangpu District in Shanghai where there are 659 intersections and 1,004 road segments. We use the 300-RSU deployment as described in Section 4.2 (see Fig. 1 in Appendix A, available in the online supplemental material). Then, we record the actual trajectories of 521 taxies according to the RSU deployment and the GPS reports of these taxies collected on 1 February 2007. We define an event as a period of 1 hour and truncate each trajectory into 24 subtrajectories. Thus, according to 24 hours in a day, we have 24 sets of subtrajectories. Both honest vehicles and malicious vehicles are randomly chosen from a subtrajectory set of 1 hour. According to our strategies to eliminate Sybil "communities" in Section 4.4.3 and the security analysis in Section 5.1, a malicious vehicle tries to provide as many Sybil trajectories that are longer than the average length of all actual trajectories as possible. If the length of the actual trajectory of the malicious vehicle $|\mathcal{T}|$ is less than the average, it provides all Sybil trajectories that are $|\mathcal{T}|-1$ long. For each subset and each simulation configuration, we run the simulation 20 times and get the average.

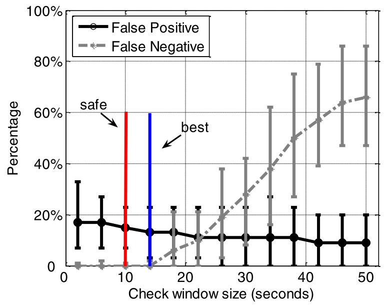

Fig. 4. Check window size versus false positive error and false negative error.

7.2 Effect of the Check Window Size

In this simulation, we examine the effect of the check window size to the system performance. We randomly choose 60 actual trajectories representing honest vehicles and 40 actual trajectories representing malicious vehicles. The trajectory length limit is set to 20 RSUs. We vary the check window size from 2 to 50 seconds with an interval of 4 seconds.

Fig. 4 plots the false positive error and false negative error as functions of the check window size. It can be seen that the false positive error decreases as the size of check window increases whereas the false negative error increases. This is because, with a larger check window, two actual trajectories have more opportunities to distinguish each other by having a negative similarity. Due to the same reason, two forged trajectories can also be falsely identified as two distinct trajectories as long as the check window is larger than the time period for a malicious vehicle to traverse any pair of RSUs contained in its actual trajectory.

Notice that if the check window is set no larger than the traverse time limit (i.e., the shortest time to travel between any pair of RSUs, 10 seconds in this RSU deployment) marked as "safe" in Fig. 4, the false negative error can reach the minimum level. Nevertheless, if a properly larger check window is adopted, the system can achieve the best performance in terms of minimizing the sum of the false positive error and the false negative error (denoted as "best" in Fig. 4).

7.3 Effect of the Trajectory Length Limit

In this simulation, we examine the effect of the trajectory length limit. We choose vehicles and generate forged trajectories in the same way as described in the above simulation. We set the check window equal to 14 seconds ("best" check window size) and vary trajectory length limit from 2 to 40 with an interval of four.

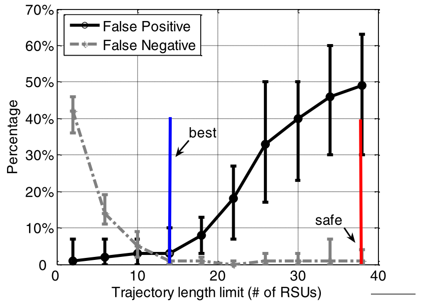

Fig. 5. Trajectory length limit versus false positive error and false negative error.

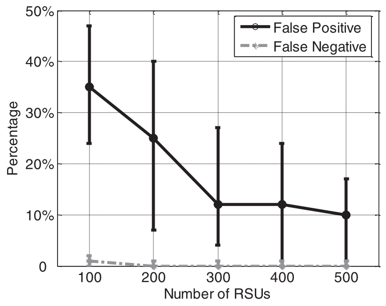

Fig. 6. RSU deployment versus false positive error and false negative error.

Fig. 5 plots the false positive error and false negative error as functions of the trajectory length limit. It can be seen that the false positive error increases as the trajectory length limit increases whereas the false negative error drops. The reason is obvious that a larger limit will treat two actual trajectories as two suspicious with an insufficient number of authorized messages. Because of the same reason, two forged trajectories can hardly be directly regarded as two actual trajectories during the exclusion test as the condition is more difficult to reach.

We can also notice from the figure that setting a properly smaller limit can achieve best performance rather than using the safe limit 40 RSUs (the maximum number of signatures a vehicle can obtained). Using the best check window size together with the best trajectory length limit, we can achieve minimum false positive error and minimum false negative error of 3 and 1 percent, respectively.

7.4 Impact of Infrastructure Deployment

In this simulation, we examine the impact of the RSU deployment. We choose vehicles and generate forged trajectories in the same way as described in the first simulation. We vary the number of RSUs deployed in the area from 100 to 500 with an interval of 100. The check window size and the trajectory length limit are set to the corresponding safe values in each deployment. The results are shown in Fig. 6.

Since we take safe trajectory length and safe check window size for each RSU deployment, the false negative error is small for all RSU deployments whereas the false positive error is relatively large. This indicates that Footprint can guarantee only a very small number of forged trajectories (less than 2 percent) can succeed at the cost of sacrificing a relatively number of honest vehicles (about 10 percent when the RSU deployment is dense).

From Fig. 6, we find that, in the early developing stage, deploying more RSUs can rapidly improve the system performance. As the distribution of RSUs is dense enough, putting more RSUs in the system will help but not that much. In this simulation setting, install RSUs at half intersections is most cost efficient.

8 Conclusion and Future Work

In this paper, we have developed a Sybil attack detection scheme Footprint for urban vehicular networks. Consecutive authorized messages obtained by an anonymous vehicle from RSUs form a trajectory to identify the corresponding vehicle. Location privacy of vehicles is preserved by realizing a location-hidden signature scheme. Utilizing social relationship among trajectories, Footprint can find and eliminate Sybil trajectories. The Footprint design can be incrementally implemented in a large city. It is also demonstrated by both analysis and extensive trace-driven simulations that Footprint can largely restrict Sybil attacks and can enormously reduce the impact of Sybil attacks in urban settings (above 98 percent detection rate).

With the proposed detection mechanism having much space to extend, we will continue to work on several directions. First, in Footprint, we assume that all RSUs are trustworthy. However, if an RSU is compromised, it can help a malicious vehicle generate fake legal trajectories (e.g., by inserting link tags of other RSUs into a forged trajectory). In that case, Footprint cannot detect such trajectories. However, the corrupted RSU cannot deny a link tag generated by itself nor forge link tags generated by other RSUs, which can be utilized to detect a compromised RSU in the system. In future work, we will consider the scenario where a small fraction of RSUs are compromised. We will develop cost-efficient techniques to fast detect the corruption of an RSU. Second, we will delve into designing better linkable signer-ambiguous signature schemes such that the computation overhead for signature verification and the communication overhead can be reduced. Last, we will validate our design and study its performance under real-complex environments based on our ongoing realistic prototype testbed built at Xi'an Jiao Tong University. Improvements will be made based on the realistic studies before it comes to be deployed in large-scale systems.

Acknowledgments

This research was supported in part by National Natural Science Foundation of China under Grant No.60933003, and IBM Joint Project.

References

[1] Y. Sun, R. Lu, X. Lin, X. Shen, and J. Su, "An Efficient Pseudonymous Authentication Scheme with Strong Privacy Preservation for Vehicular Communications," *IEEE Trans. Vehicular Technology,* vol. 59, no. 7, pp. 3589-3603, Sept. 2010.

[2] R. Lu, X. Lin, H. Zhu, and X. Shen, "An Intelligent Secure and Privacy-Preserving Parking Scheme through Vehicular Communications," *IEEE Trans. Vehicular Technology,* vol. 59, no. 6, pp. 2772-2785, July 2010.

[3] J.R. Douceur, "The Sybil Attack," *Proc. First Int'l Workshop Peer-to-Peer Systems (IPTPS '02),* pp. 251-260, Mar. 2002.

[4] J. Eriksson, H. Balakrishnan, and S. Madden, "Cabernet: Vehicular Content Delivery Using WiFi," *Proc. MOBICOM '08*, pp. 199-210, Sept. 2008.

[5] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D.S. Wallach, "Secure Routing for Structured Peer-to-Peer Overlay Networks," *Proc. Symp. Operating Systems Design and Implementation (OSDI '02)*, pp. 299-314, Dec. 2002.

[6] B. Dutertre, S. Cheung, and J. Levy, "Lightweight Key Management in Wireless Sensor Networks by Leveraging Initial Trust," Technical Report SRI-SDL-04-02, SRI Int'l, Apr. 2002.

[7] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil Attack in Sensor Networks: Analysis & Defenses," *Proc. Int'l Symp. Information Processing in Sensor Networks (IPSN '04)*, pp. 259-268, Apr. 2004.

[8] S. Capkun, L. Buttyań, and J. Hubaux, "Self-Organized Public Key Management for Mobile Ad Hoc Networks," *IEEE Trans. Mobile Computing*, vol. 2, no. 1, pp. 52-64, Jan.-Mar. 2003.

[9] C. Piro, C. Shields, and B.N. Levine, "Detecting the Sybil Attack in Mobile Ad Hoc Networks," *Proc. Securecomm and Workshop*, pp. 1-11, Aug. 2006.

[10] N. Borisov, "Computational Puzzles as Sybil Defenses," *Proc. Sixth IEEE Int'l Conf. Peer-to-Peer Computing (P2P '06)*, pp. 171-176, Oct. 2006.

[11] P. Maniatis, D.S.H. Rosenthal, M. Roussopoulos, M. Baker, T. Giuli, and Y. Muliadi, "Preserving Peer Replicas by Rate-Limited Sampled Voting," *Proc. 19th ACM Symp. Operating Systems Principles (SOSP '03)*, pp. 44-59, Oct. 2003.

[12] H. Yu, M. Kaminsky, P.B. Gibbons, and A. Flaxman, "Sybilguard: Defending against Sybil Attacks via Social Networks," *Proc. SIGCOMM*, pp. 267-278, Sept. 2006.

[13] M.S. Bouassida, G. Guette, M. Shawky, and B. Ducourthial, "Sybil Nodes Detection Based on Received Signal Strength Variations within Vanet," *Int'l J. Network Security*, vol. 9, no. 1, pp. 22-32, 2009.

[14] B. Xiao, B. Yu, and C. Gao, "Detection and Localization of Sybil Nodes in Vanets," *Proc. Workshop Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS '06)*, pp. 1-8, Sept. 2006.

[15] T. Zhou, R.R. Choudhury, P. Ning, and K. Chakrabarty, "Privacy-Preserving Detection of Sybil Attacks in Vehicular Ad Hoc Networks," *Proc. Fourth Ann. Int'l Conf. Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous '07)*, pp. 1-8, Aug. 2007.

[16] Q. Wu, J. Domingo-Ferrer, and U. Gonźalez-Nicolás, "Balanced Trustworthiness, Safety and Privacy in Vehicle-to-vehicle Communications," *IEEE Trans. Vehicular Technology*, vol. 59, no. 2, pp. 559-573, Feb. 2010.

[17] L. Chen, S.-L. Ng, and G. Wang, "Threshold Anonymous Announcement in VANETs," *IEEE J. Selected Areas in Comm.*, vol. 29, no. 3, pp. 1-11, Mar. 2011.

[18] C. Chen, X. Wang, W. Han, and B. Zang, "A Robust Detection of the Sybil Attack in Urban Vanets," *Proc. IEEE Int'l Conf. Distributed Computing Systems Workshops (ICDCSW '09)*, pp. 270-276, June 2009.

[19] S. Park, B. Aslam, D. Turgut, and C.C. Zou, "Defense against Sybil Attack in Vehicular Ad Hoc Network Based on Roadside Unit Support," *Proc. 28th IEEE Conf. Military Comm. (MILCOM '09)*, pp. 1-7, Oct. 2009.

[20] IEEE Vehicular Technology Soc.: 5.9 GHz Dedicated Short Range Comm. (DSRC) - Overview. http://grouper.ieee.org.groups/scc32/dsrc/, 2011.

[21] J.K. Liu, V.K. Wei, and D.S. Wong, "Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups (Extended Abstract)," *Proc. Ninth Australasian Conf. Information Security and Privacy (ACISP '04)*, pp. 325-335, 2004.

[22] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup, "Anonymous Identification in Ad Hoc Groups," *Proc. Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT '04)*, pp. 609-626, 2004.

[23] P.P. Tsang and V.K. Wei, "Short Linkable Ring Signatures for E-Voting, E-Cash and Attestation," *Proc. Information Security Practice and Experience Conf. (ISPEC '05)*, pp. 48-60, 2005.

[24] P.P. Tsang, V.K. Wei, T.K. Chan, M.H. Au, J.K. Liu, and D.S. Wong, "Separable Linkable Threshold Ring Signatures," *Proc. Int'l Conf. Cryptology in India (INDOCRYPT '04)*, pp. 384-398, 2004.

[25] P.R. Östergård, "A Fast Algorithm for the Maximum Clique Problem," *Discrete Applied Math.*, vol. 120, nos. 1-3, pp. 197-207, 2002.

Shan Chang received the BS degree in computer science and Technology from Xián Jiaotong University in 2004, where she is currently working toward the master-doctoral degree in the Department of Computer Science and Technology. Her research interests include security and privacy in mobile networks and wireless sensor networks.

Yong Qi received the PhD degree in computer science and technology from Xián Jiaotong University in 2001. He is a full professor in the Department of Computer Science and Technology of Xián Jiaotong University. His research interests include sensor networks, operating system, distributed middleware, and services computing. He is a member of the IEEE.

Hongzi Zhu received the PhD degree in computer science from Shanghai Jiao Tong University in 2009. He is an assistant professor in the Department of Computer Science and Engineering at the Shanghai Jiao Tong University. His research interests include mobile networks, social networks and network security. He is a member of the IEEE and IEEE Computer Society and Communication Society.

Jizhong Zhao received the BS and MS degrees in the Mathematic Department from Xi'an Jiaotong University. He received the PhD degree in computer science, focusing on Distributed System, from Xi'an Jiaotong University in 2001. He is a professor of Computer Science and Technology Department, Xi'an Jiaotong University. His research interests include computer software, pervasive computing, distributed systems, and network security. He is a member of the IEEE and IEEE Computer society, and a member of the ACM.

Xuemin (Sherman) Shen received the BS degree from Dalian Maritime University, China, in 1982, and the MSc and PhD degrees from Rutgers University, New Jersey, in 1987 and 1990, respectively, all in electrical engineering. He is a professor and University research chair, Department of Electrical and Computer Engineering, University of Waterloo. His research focuses on mobility and resource management, UWB wireless networks, wireless network security, and vehicular ad hoc and sensor networks. He served as an area editor for *IEEE Transactions on Wireless Communications* and editor-in-chief for Peer-to- Peer Networks and Applications. He is a distinguished lecturer of the IEEE Communications Society, fellow of the IEEE, and fellow of the Engineering Institute of Canada.