# VProof: Lightweight Privacy-Preserving Vehicle Location Proofs

Yifan Zhang, Chiu C. Tan, *Member, IEEE*, Fengyuan Xu, *Member, IEEE*, Hao Han, and Qun Li, *Senior Member, IEEE*

***Abstract*—Transportation systems relying on vehicles to collect data for services such as road condition monitoring are vulnerable to malicious vehicles injecting large amounts of fake data. A particularly serious type of attack is one in which attackers report fake information about numerous places without actually having been there. In this paper, we present VProof, which is a vehicle location proof scheme that enables a vehicle to prove that its location claims match its historical locations. With VProof, vehicles construct their location proofs by simply extracting relevant contents from the packets received from roadside units. Our scheme is lightweight, since there is no communication required for a prover to obtain a location proof. Our scheme also well preserves users' privacy, as we do not put any information that can be related to a user's ID in a location proof. We have implemented a prototype VProof system and evaluated it with extensive real-world experiments. Our evaluation results show that VProof is able to reliably prove vehicle's locations without leaking any user privacy.**

***Index Terms*— Location proof, privacy, security, vehicular networks.**

## I. Introduction

UBIQUITOUS computing technologies, such as sensing and wireless communication, have enabled intelligent transportation systems (ITS), with which people can achieve safer and more efficient everyday transportation. In today's ITS, a popular category of applications is that vehicles report information about the transportation system elements (e.g., drivers and road conditions) to the ITS system for services such as real-time traffic control and roads maintenance [1], [2]. However, before accepting data about a location reported by a vehicle, ITS operators need to verify if the vehicle visited the location at the time indicated in the reported data. Failing to do so will allow malicious users to launch an attack to the ITS system by reporting fake information about places where he did not actually visit. The damages of the attack are particularly serious, since the attacker can report fake information about numerous places by just clicking mouse at home.

To verify whether a vehicle's location claims match its actual historical locations, ITS operators need a location proof scheme featuring the following properties. *First*, the location proof should be lightweight. This property is extremely important in vehicular environments, since location proof issuers may need to issue location proofs to tens or hundreds of vehicles on a busy road at the same time. *Second*, the location proof needs to well preserve users' location privacy. Concerns about users' (we will use "user" and "vehicle" alternately) location privacy have become major considerations when deploying location-related services in ITS systems [3]. Car owners can simply opt out of providing *any* data if their privacy is threatened. *Third*, the location proof scheme needs to be able to generate fine-grained location proofs, because the locations reported in the user collected information (e.g., there is a pothole somewhere on the road) have fine granularity.

To detect malicious users who report bogus data, the conventional solution is to assign each vehicle with some cryptographic keys. Each vehicle will sign each piece of data with its secret key before uploading to the ITS system. The idea is that by having a means to track back users, the amount of bogus data will be reduced, since malicious users do not want to be caught. Similar schemes have also been proposed to protect the privacy of honest users to encourage participation [4]–[6]. Nonetheless, these solutions all require deploying a large-scale public-key infrastructure (PKI) scheme to associate specific keys with individual vehicles. The use of the large-scale PKI scheme usually requires complicated management of digital certificates, as well as complex coordination between the administrative authority and participating users. While the PKI-based approach may be possible in theory, for instance, a PKI administered by a local DMV, it is less clear if this will be done in practice, particularly in a large country. We can point to the difficulties in getting the different states in the United States to standardize on a common driver's license as evidence of the impracticality of a widely deployed PKI solution. Thus, a solution that does *not* rely on such large-scale infrastructure to provide privacy protections is needed.

The idea of location proofs has been considered by other types of applications before. The general approach is to let certain authorized entities with fixed geolocations perform as location proof issuers. The location proof issuers issue location proofs, which are unique and unforgeable, to nearby location provers, who need to prove their historical locations to a location verifier later. A location prover is believed to

Manuscript received April 17, 2013; revised October 14, 2013; accepted January 2, 2014. Date of publication May 2, 2014; date of current version January 13, 2015. This work was supported in part by the National Science Foundation under Grant CNS-1320453 and Grant CNS-1117412 and in part by CAREER Award CNS-0747108. The review of this paper was coordinated by Prof. Y. Zhang.

Y. Zhang and Q. Li are with the Department of Computer Science, College of William and Mary, Williamsburg, VA 23187-8795 USA (e-mail: yzhang@cs.wm.edu; liqun@cs.wm.edu).

C. C. Tan is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: cct@temple.edu).

F. Xu is with the Storage System Group at NEC Laboratories America, Princeton, NJ 08540 USA (e-mail: fxu@nec-labs.com).

H. Han is with the Networks and Security Group in Intelligent Automation, Inc., Rockville, MD 20855 USA (e-mail: hhan@i-a-i.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2014.2321666

0018-9545 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

be in the vicinity of a proof issuer at a certain time if the prover possesses valid location proofs [7]–[9]. We cannot apply the same approach toward vehicular environments because the location proof granularity achieved by the existing location proof solutions is coarse: They can only prove that at a certain time a user was within the communication radius of a proof issuer (e.g., the communication range of a wireless AP), but not at a finer granularity. This allows a malicious user to *statically* collect the location proofs issued by a proof issuer and report fake information about places where he never visited but are within the proof issuer's communication radius. Meanwhile, the existing solutions require a proof issuer to perform multiple rounds of interactions with a prover to issue a location proof specifically to the prover, which is not scalable to vehicular environments, where lots of vehicles may be requesting location proofs at the same time. More discussion on the limitations of the existing location proof solutions to be applied in vehicular environments can be found in the full version of [10].

We propose VProof, which is a vehicular location proof solution designed without using PKI schemes. In our solution, RSUs (roadside units), which are necessary components of ITS systems [1], [2], continuously broadcast packets that are specifically for the location proof functionalities. We name these packets as *VPackets*. We observe that the received signal strength (RSS) of the series of VPackets received by a vehicle when it passes an RSU exhibit similar patterns over time if the transmission power of the VPackets is fixed (we will elaborate on this observation later in Section IV). In VProof, if a user claims his data were collected at a certain place, we require him to show that he has seen the correct RSS pattern of the VPackets sent by a nearby RSU. With our scheme, the correct RSS pattern can only be obtained if the user indeed drove past the claimed place. Thus, we achieve fine-grained location proofs and prevent malicious users from statically collecting location proofs. Our scheme has the following unique properties compared with the related work. *First*, our scheme is *lightweight*. In VProof, RSUs (proof issuers) issue location proofs by simply broadcasting packets, and vehicles (provers) construct location proofs by extracting relevant contents from the RSU packets they received. This makes our scheme scalable in busy vehicular environments where lots of vehicles may be requesting location proofs at the same time. *Second*, the VProof scheme is *privacy preserving*, because it puts no information that can be related to a user's identification. *Third*, the VProof scheme can provide *fine-grained* and unforgeable location proofs.

Although using RSS patterns to design a location proof solution for vehicular environment seems promising at first glance, it brings three major challenges as summarized in [10]. Our contributions can be summarized as follows.

- We propose VProof, which is a lightweight and privacy-preserving location proof solution that verifies if a vehicle's location claims match its historical locations. To the best of our knowledge, VProof is the first location proof solution designed for vehicular environments and the first that does not rely on PKI to achieve the functionalities of location proofs.
- We design an efficient algorithm that can reliably determine if two series of packet RSS are similar given potential packet losses and inaccurate RSS measurements.
- We implement a prototype of the VProof system and evaluate the prototype system with extensive experiments performed on real road conditions.

## II. Related Work

To prevent users from submitting fake information in vehicular networks, the existing solutions typically use anonymous authentication. Work by Xi *et al.* [4] proposes a symmetric random key-set scheme, where each vehicles possesses a set of symmetric keys randomly chosen from a key pool, to authenticate vehicles into vehicular networks. The scheme proposed by Calandriello *et al.* [5] addresses the problem of anonymous message authentication using asymmetric keys and group signature in vehicular networks. The efficient conditional privacy preservation protocol (ECPP) [6] achieves anonymous message authentication under the help of its on-the-fly short-time anonymous keys between vehicles and RSUs.

The reasoning behind the existing approaches is that if car owners are aware that bogus data can be traced back to them, they will not intentionally upload incorrect data. However, car owners can simply opt out of providing any data if their privacy is threatened. Our work provides a technique to enable a user to prove that his historical locations are in accordance with the data he reports. This allows ITS operators to prevent the attacks where users report information about places they did not visit. Our approach achieves strong user privacy protection, since we do not place any information regarding the user's identity nor link any cryptographic keys with the user, and thus, there is no way users reporting data to the ITS systems can be traced. We argue that our approach is more suitable in situations where user privacy outweighs other concerns.

Concerns about users' location privacy have become major considerations when deploying location-related services in ITS systems [3]. Existing solutions use group navigation and dynamic pseudonyms [11], mix zones, and vehicular mix networks [12] or group communication [13] to defend users' location privacy. VProof also well protects users' location privacy since we do not place any information regarding the user's identity nor link any cryptographic keys with the user.

Changing wireless packet transmission power and measuring the RSS have recently been used in localizations [14], [15] and rogue vehicular AP detection [16], [17]. All these existing works rely on accurate RSS measurements to determine the distance between two communicating wireless nodes. In our work, we have no dependence on accurate RSS measurements for the following two reasons. *First*, changing packet transmission power and measuring RSS in our solution are *not* to determine the distance between two wireless nodes. Instead, we use them to hide the inherent RSS patterns from users. Thus, our work does not need as accurate RSS readings as the existing works do. *Second*, our RSS pattern similarity comparison algorithm is specifically designed to cope with inaccurate RSS measurements and packet losses.

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

## III. Background

In ITS data collection applications, vehicles sense and collect data about surrounding elements by their add-on vehicle sensors. The sensed data, along with the corresponding *metadata* that tell when and where the sensed data were generated, are uploaded to backend servers located at the infrastructure side, either immediately after they are generated or sometime later when appropriate [18]–[20]. The ITS systems operate vehicular networks that enable wireless communication between vehicles and the infrastructure via RSUs. We assume that RSUs are controlled by the ITS operator, and all vehicles are equipped with GPS receivers.

We consider the threat that malicious users target at disrupting ITS systems by reporting fake information about numerous places where they did not actually visit. Please note that we do not consider the threat that a malicious user physically presents at a place and report fake information about it, as we deem this threat has much less impacts than the one we are considering (see full version of the paper [10] for more discussion). If there is no scheme to allow ITS operators to verify whether the reporting users have actually visited the places indicated in the reported data, a malicious user can easily generate and report bogus data about lots of places without actually visiting those places. The amount of the bogus data could overwhelm that of the honest data. Existing works for filtering abnormal data in vehicular networks [21], [22] do not work in this case, because they hold an assumption that the amount of abnormal data should not be more than that of normal data. Meanwhile, as we have previously discussed, we prefer not to use PKI to solve the problem. Thus, we ask the question: Without using PKI systems, can we provide a scheme to let ITS operators verify if a user's historical locations are in accordance with the data he submits so that they can prevent the threat we just described?

We make the following assumptions about malicious users. *First*, malicious users have the same equipment as honest users and have certain knowledge of the information about the RSUs in the ITS system, such as the ESSIDs and GPS locations of the RSUs. However, they do not know any secret keys shared between the RSUs and the ITS system. *Second*, malicious users cannot control any infrastructure units or replicate them in exactly the same way. For example, malicious users cannot replicate a certain legitimate RSU by placing the same hardware on the same roadside pole. This prevents malicious users from obtaining similar profiling data as the authority does.

## IV. Two Key Observations

Through real-world experiments, we observe that the RSS of a series of RSU packets received by a vehicle when it passes an RSU, which is continuously broadcasting packets with fixed power, exhibit similar patterns over time. In the experiments, we deployed a wireless node, which broadcast packets at a rate of 100 packets/s with full transmission power, at the roadside of a downtown environment. We drove a car past the roadside wireless node and collected its packets at different times. Fig. 1 shows the RSS patterns of two series of packets collected in the experiment, where series-A data were collected

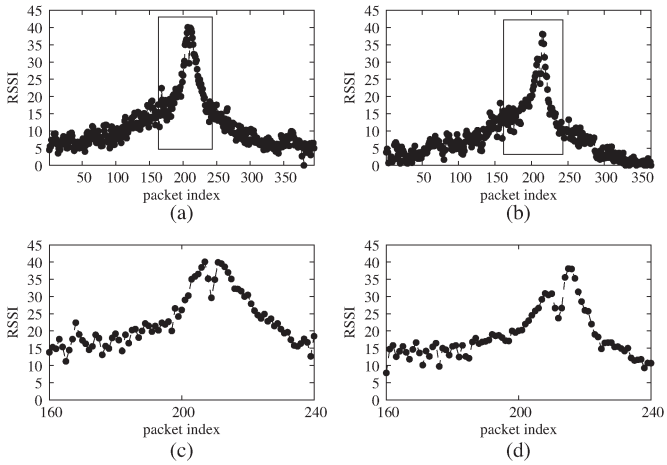



Fig. 1. RSS pattern of packet series collected at different times. (a) RSS pattern of packet series A. (b) RSS pattern of packet series B. (c) Center details of packet series A's RSS pattern. (d) Center details of packet series B's RSS pattern.

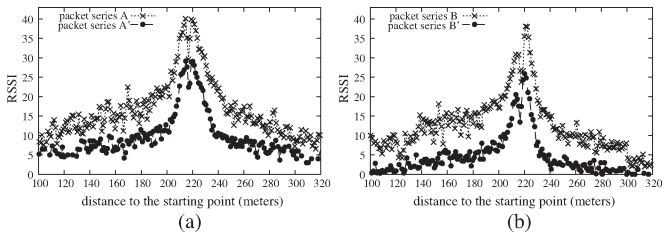



Fig. 2. RSS of two RSU packet series with different tx power values. (a) RSS patterns of packet series A and A′. (b) RSS patterns of packet series B and B′.

in the morning of one day when there were less cars on the road, and series-B data were collected during peak hours in the afternoon of another day. We can see that, although there are slight differences in RSS amplitudes and pattern shapes, which are due to factors such as different temperatures and different moving obstacles (i.e., cars and trucks) on the road, the two series of packets do exhibit similar RSS patterns.

We also observe that the RSS difference between two packets that are transmitted by the same RSU using two different power levels and are received at the same location is *roughly* a constant across the RSU's RF range over time. Fig. 2 shows the RSS of two groups of RSU packet series. Packet series A and B, which were transmitted using full power, are the same as in Fig. 1. Packet series A′ and B′, which were transmitted using half the full power, were obtained at the same times as A and B, respectively. We align the two RSU packet series obtained in the same experiment based on the distance between each packet's reception location and a fixed starting point. We can observe that the RSS difference under two different transmission power levels at the same location is roughly the same (around 8 dBm). We will quantify how stable this RSS difference is later in Section VI.

## V. VProof Design

With the two given observations, we design a location proof scheme using RSS of RSU packets. Generally, RSUs continuously broadcast packets that are specifically for the location

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

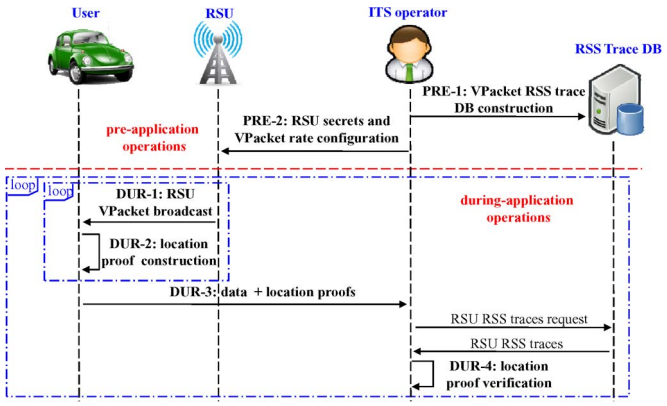



Fig. 3. Operation flow of VProof.

proof functionalities (named as "VPackets"). Each VPacket is broadcast using a *randomly* chosen transmission power. Since the transmission power is randomly selected, the RSS of the VPackets received by vehicles exhibit no pattern. Each VPacket includes certain *encrypted* information that contains the transmission power of the packet. Vehicles collect the VPackets, construct location proofs based on information in the VPackets and their own GPS readings, and submit the location proofs to the ITS system for verification. Using the information in the location proofs, specifically the transmission power of each VPacket, the ITS operators can restore the inherent VPacket RSS patterns, which are the RSS patterns if the VPackets were transmitted using full power. The location proofs are deemed as valid only if they can be used to correctly restore the inherent RSS patterns of RSUs. Since the transmission power of each VPacket is *only* known by the ITS operators, we enforce the unforgeability of the location proofs VProof constructs.

The operation flow of VProof is shown in Fig. 3. The *preapplication operations* are the operations performed before the data collection applications are deployed, and the *during-application operations* are those performed with the data collection applications.

### A. PRE-1: VPacket RSS Trace Database Construction

The RSS trace DB contains VPacket RSS traces of each RSU. In our design, there are $N$ RSS traces associated with each possible vehicle trajectory around each RSU $U_i$. Each RSS trace contains an RSS series of VPackets collected by driving a car past $U_i$ on the trajectory. During the profiling process, $U_i$ broadcasts VPackets using *full* transmission power. In the DB, assuming there are $L$ transmission power values, $U_i$ is also associated with average RSS differences between each non-full power $P_j (j = 1, \ldots, L-1)$ and full power $P_f$: $RSS_{U_i, P_f \to P_j}$. We will describe the experiment that obtains this average RSS difference in Section VI. The profiling process only needs to be done once.

### B. PRE-2: RSU Secrets and VPacket Rate Configuration

The system operator generates an RSU-specific secret $s_{U_i}$ for each RSU $U_i$ and configures it to $U_i$. The secret $s_{U_i}$ is for encrypting/decrypting the VPacket's transmission power and also for generating/verifying the VPacket authentication message (described later). For a vehicular wireless node moving in an outdoor environment, the RSS of its received packets vary a lot. To deal with RSS instability, we want users to receive $n$ VPackets within a period of coherence time, which is the time duration over which the RSS is considered to be not varying. Then, we can take the average of these $n$ VPackets' RSS as a data point in the RSS series. Hence, the VPacket rate for an RSU $U_i$ is calculated as $n/T_{\text{coherence}}$, where $T_{\text{coherence}}$ is the average coherence time of the road section around $U_i$.

### C. DUR-1: VPackets Broadcast (Proof Issuing by RSU)

*1) VPacket Transmission Power Selection:* We design an algorithm to allow an RSU $U_i$ to randomly select a transmission power $p$ from $L$ options: $P_f, P_1, \ldots, P_{L-1}$, with $P_f$ being the full power and $P_1, \ldots, P_{L-1}$ being the other $L-1$ none-full power levels. The algorithm ensures that the RSU would broadcast VPackets using the same power for a period of time that is larger than coherence time $T_{\text{coherence}}$. The details of the algorithm is shown in [10]. After the transmission power $p$ is determined, $U_i$ encrypts $p$ using a symmetric-key algorithm $SE$, with the combination of $s_{U_i}$ and time $t$ (i.e., the time when the VPacket is generated) as the cryptographic key

$$C_p \leftarrow SE_{s_{U_i}, t}(p). \tag{1}$$

*2) VPacket Authentication Message Generation:* To verify if the location proofs are constructed based on authentic VPackets, an RSU $U_i$ generates a *VPacket authentication message (VAM)* for each VPacket as

$$VAM \leftarrow H\left(U_i, s_{U_i}, t, C_p\right) \tag{2}$$

where $H$ is a cryptographic hash function (e.g., MD5 and SHA-1) that hashes $U_i$ (the ID of the RSU), $s_{U_i}$ (the secret of the RSU), $t$ (the time when the VPacket is generated), and $C_p$ (the ciphertext of the transmission power $p$) into a single piece of message.

Finally, the RSU $U_i$ puts $VAM$, $C_p$, and $t$ into the VPacket and broadcasts the VPacket using power $p$.

### D. DUR-2, 3: Location Proof Construction and Submission (by Vehicles)

Upon receiving a VPacket, a vehicle constructs a location proof $(LP)$ as

$$LP = \langle U_i, \quad VAM, \quad t, \quad RSS, \quad C_p, \quad LOC \rangle \tag{3}$$

where 1) $U_i$ is the ID of the VPacket's originating RSU, 2) $VAM$ is the VPacket authentication message, 3) $t$ is the time when the VPacket is generated, 4) *RSS* is the RSS of the VPacket, 5) $C_p$ is the ciphertext of the VPacket transmission power $p$, and 6) *LOC* is the vehicle's GPS location when the packet is received. The first five items are extracted from the VPacket, and the last item is obtained from the vehicle's onboard GPS device.

When the user uploads newly collected data to the ITS system, he also uploads all the location proofs constructed since the last data submission to the ITS system.

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

**Algorithm 1:** User RSS series $S_{user}$ construction

**Data**: Raw RSS sequence: $S_{raw} = u_1 \cdots u_m$;
VPacket generation time sequence: $T_{raw} = t_1 \cdots t_m$;
Deciphered power sequence: $P_{raw} = q_1 \cdots q_m$.
**Result**: User RSS series: $S_{user} = v_1 \cdots v_n$;
Transmission Power sequence: $P_{S_{user}} = p_1 \cdots p_n$.

1 $S_{user} \leftarrow \phi$, $P_{S_{user}} \leftarrow \phi$;
2 **for** $i \leftarrow$ to $m$ **do**
3  **if** $S_{user} == \phi$ **then**
4  $time \leftarrow t_i$; $power \leftarrow q_i$; $pool \leftarrow \{u_i\}$;
5  **else**
6  **if** $(t_i > time + T_{coherence})$ *or* $(q_i \neq power)$ **then**
7  Add $average(pool)$ at the end of $S_{user}$;
8  Add $power$ at the end of $P_{S_{user}}$;
9  $time \leftarrow t_i$; $power \leftarrow q_i$; $pool \leftarrow \{u_i\}$;
10  **else**
11  Add $u_i$ to $pool$;
12 **return** $S_{user}$, $P_{S_{user}}$;

### *E. DUR-4: Location Proof Verification (by ITS Operators)*

The ITS operator divides the location proofs received from the *same* upload connection into batches such that each batch of location proofs share the same RSU ID. Then, the operator verifies the location proofs batch by batch. A valid batch of location proofs with RSU ID $U_i$ indicate that the proof submitter has actually driven past $U_i$ at the time indicated in the location proofs.

*1) VPacket Authentication Message Verification:* Given a batch of $m$ location proofs, the ITS operator verifies the $VAM$ contained in each location proof as follows. He computes the message content by using (2) with the parameters $U_i$, $t$, $C_p$, and $s_{U_i}$, where $U_i$, $t$, and $C_p$ are extracted from the location proof, and $s_{U_i}$ is kept by the operator. If the computed content is different from the $VAM$ contained in the location proof, $VAM$ is deemed invalid. An invalid $VAM$ indicates that at least one parameter of $U_i$, $t$, and $C_p$ provided in the location proof has been tampered with. If there exists one location proof containing invalid $VAM$, the whole batch of location proofs is invalid. Note that a batch of location proofs with valid $VAM$s does not necessarily mean that the batch of location proofs is valid, because a malicious user can *statically* collect VPackets, obtain valid $VAM$s, and present them in the location proofs.

*2) User RSS Series Construction:* Given a batch of $m$ location proofs with valid $VAM$s, the operator constructs a user RSS series $S_{\text{user}}$ according to Algorithm 1. In the algorithm, $S_{\text{raw}}$ is the sequence of RSS in the $m$ location proofs, $T_{\text{raw}}$ is the corresponding sequence of VPacket generation times, and $P_{\text{raw}}$ is the corresponding sequence of deciphered transmission power levels. The algorithm outputs the user RSS series $S_{\text{user}}$, in which each data point is computed as the *average* RSS (line 7) of the VPackets that were transmitted under the same power and within a period of coherence time (line 6). The algorithm also outputs $P_{S_{\text{user}}}$, which is a sequence of deciphered transmission power levels. Each data point of $P_{S_{\text{user}}}$ corresponds to an RSS data point in $S_{\text{user}}$ (line 8).

*3) User RSS Series Preprocessing:* The preprocessing of the user RSS series $S_{\text{user}}$ has two goals. The first is to make $S_{\text{user}}$ location even. In real road situations, users may stop on the road for a while (due to red lights) or drive with a speed that is far less than the speed limit (due to congested traffic). In these cases, $S_{\text{user}}$ will contain much more data points measured around some locations than from other locations. Our algorithm tunes $S_{\text{user}}$ by removing the redundant points based on the parameters of *LOC* and $t$ presented in the corresponding location proofs. The second goal of the preprocessing is to identify the user's trajectory based on which the operator can select the corresponding RSS traces from the RSS DB. The trajectory identification is based on the GPS locations contained in the proofs.

*4) User RSS Series Pattern Restoration:* The user RSS series $S_{\text{user}} = v_1 \cdots v_n$ has no pattern since the VPackets were broadcast using random transmission power. Therefore, before comparing the patterns of user-submitted RSS series and the profiled RSS series stored in the trace DB, the ITS operator needs to restore $S_{\text{user}}$'s inherent RSS pattern (i.e., the RSS pattern if all the VPackets were transmitted using full power). This is accomplished by adding $RSS_{U_i, P_f \rightarrow P_j}$ (recall that this information is associated with $U_i$ in the RSS DB) to each data point $v_k \in S_{\text{user}}$, if $p_k \in P_{S_{\text{user}}}$ is equal to $P_j$, where $1 \leq j \leq L-1$ ($L$ is the number of transmission power levels) and $1 \leq k \leq n$.

*5) User RSS Series Validation:* To validate the user RSS series $S_{\text{user}}$, the ITS operator fetches the $N$ RSS traces associated with the user trajectory from the RSS trace DB and derives $N$ DB RSS series $S_{DB,i}$ ($i \in [1, N]$) in the same way as constructing $S_{\text{user}}$. The ITS operators compares the similarity between the user RSS series $S_{\text{user}}$ and each of the $N$ profiled RSS traces, respectively. The user RSS series is deemed valid if there are enough amounts of matches.

It is difficult to decide if two RSS series have similar patterns for the following three reasons. *First*, different hardwares may have different readings on the same received packet, because they may have different noise floors. To address this issue, our RSS series comparison algorithm is designed to compare patterns of the *quantized* series, which are not impacted by amplitudes of the RSS readings. *Second*, RSS measurements are sensitive to many factors particularly in an outdoor moving environment. *Third*, due to busy vehicular wireless environment, vehicles may not received all the VPackets based on which the location proofs are constructed. To address the second and the third challenges, we designed a *dynamic time warping* (DTW)-based [23], [24] algorithm to compare two RSS series.

Our RSS series similarity comparison algorithm compares the user RSS series $S_{\text{user}}$ with each of the $N$ DB RSS series $S_{DB,i}$ according to the following steps.

The *first* step is to quantize both the RSS series $S_{\text{user}}$ and $S_{DB,i}$ using a $K$-number alphabet. The goal of the RSS series quantization is to, as pointed out previously, remove the factors that can cause different amplitudes on RSS readings (e.g., hardware differences). The quantized value of each data point in a RSS series reflects the position of data point's value within the value range of the RSS series. We use a simple example to illustrate how the quantization process works. Suppose that there is an RSS series whose RSS values are in the range of [1, 30], and we want to quantize them using a three-number alphabet {0, 1, 2} (i.e., $K = 3$). Then, our algorithm converts the data points in the RSS series with values from in the ranges of [1, 10], [11, 20], and [21, 30] to numbers 0, 1, and 2, respectively.

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

The *second* step is to obtain the warped versions of the quantized RSS series. The goal of this step is to cope with inaccurate RSS measurements and potential missing data points due to losses of VPackets. In this step, both quantized $S_{\text{user}}$ and quantized $S_{DB,i}$ are converted to their corresponding warped versions using a dynamic time warping (DTW) based algorithm. Due to page limitations, we present the details of the algorithm in [10].

In the *final* step, the algorithm calculates the similarity score between the two RSS series, based on which a conclusion is drawn. Suppose the length of the two equal-length quantized warped RSS series $S'_{\text{user}}$ and $S'_{DB,i}$ is $l$, the similarity score between $S'_{\text{user}}$ and $S'_{DB,i}$ is defined as

$$sim\_score = 1 - \frac{hamming\_dist\left(S'_{\text{user}}, S'_{DB,i}\right)}{l} \quad (4)$$

where the function $hamming\_dist(\cdot)$ calculates the hamming distance, which is the number of different bits, between the two input sequences. If the score is higher than a threshold $s_{\text{thresh}}$, the user RSS series $S_{\text{user}}$ is considered to be similar to the $i$th DB RSS series $S_{DB,i}$. We will empirically identify the suitable value of $s_{\text{thresh}}$ later in Section VI. Within the comparisons between $S_{\text{user}}$ and the $N$ DB RSS series, if there are no less than $n_{\text{thresh}} \cdot N$ matches, the batch of location proofs are identified as valid.

### F. Threat Prevention Analysis and Discussion

We analyze how VProof achieves privacy-preserving location proof and discuss more related issues in the full version of [10].

## VI. Evaluation

### A. Implementation and Experimental Setup

*1) Prototype System:* Our prototype VProof system consists of a wireless access point (Wiligear WBD-500 integrated radio board) mounted at the roadside that serves as an RSU, a vehicular node equipped with a wireless receiver (Lenovo T61 + wireless card + external omnidirectional antenna) that acts as ITS system users and a backend server (Dell T3500) that performs the operations done by the operator. The wireless AP runs a program that controls VPacket transmission power and broadcast rate according to the VProof scheme. The vehicular wireless node runs a program that constructs location proofs when VPackets arrive. To study whether hardware differences at the user side have impacts on our scheme, we have used two different wireless cards on the vehicular wireless node, i.e., Ubiquiti SWX-SRC and Wistron CB9-GP, both of which have an external antenna socket. The backend server processes the location proofs constructed by the vehicular node offline. Note that in real ITS data collection applications, the sensed data and metadata are not required to be uploaded to the backend server in real time [18]–[20]. Users can choose to upload the data anytime they feel appropriate, for example, when the uploads will not contend with other important tasks. Therefore, our choice of letting the backend server process the location proofs offline conforms to the reality.

*2) Per-Packet Transmission Power Control:* The program we ran on the wireless AP is able to change the transmission power for each packet. This per-packet power control is achieved by specifying the desired transmission power in the packet's radiotap structure. With the per-packet transmission power control, it is possible to change VPacket's transmission power randomly while not affecting the normal tasks done by the RSU (e.g., beacon broadcasting).

*3) Experimental Setup:* We have conducted extensive experiments on real road situations to evaluate our solution. The experiments were performed at three different locations in downtown environment with busy road traffic. We built the RSS DB by profiling the RSU RSS pattern (i.e., we drove a car past the RSU, which was broadcasting VPackets using full transmission power, and collected the VPackets) at the three locations in one day. We then performed experiments to collect user RSS traces (i.e., we drove a car past the RSU, which was broadcasting VPackets using randomly selected transmission power levels, and collected the VPackets) at each location in three other days. Three RSU transmission power levels were used: full power, 3/4 power, and half power. Note that all the three tested cases involved relatively straight routes. We expect for our scheme to work similarly for cases containing turnings and other irregular routes. This is because the VPacket RSS pattern of a particular route is relatively stable as long as the surrounding structures remain unchanged. Furthermore, the driving speed in our experiment was around 20 to 30 mi/h. We expect for our solution to also work in a high-speed scenario. This is because the VPacket rate by RSUs is dynamically adapted based on the average road speed: recall that the VPacket rate is calculated based on the coherence time of the environment, which is closely related to speed limit of the road.

### B. Experimental Results

*1) Coherence Time Measurement:* We measured the coherence time of the wireless channel at each of the three experiment locations as Camp and Knightly [25] did. Specifically, we let the RSU broadcast small packets (100 bytes/packet) at a very high packet rate (500 packets/s). Then, we measured the RSS differences between different sizes of VPacket windows based on which we determined the coherence time of the channel. We found that the coherence times at our experiment locations were around 100 ms. Thus, in our experiments, we set the VPacket rate to 100 frames/s, as we wanted users to receive ten VPackets per period of coherence time. Meanwhile, according to Algorithm 1, when constructing the RSS series, we computed an RSS data point as the average RSS of VPackets that are transmitted under the same power and within 100 ms.

*2) Quantifying the Stableness of RSS Difference of VPackets With Two Different Power Levels:* To restore the inherent RSS pattern from the RSS of VPackets broadcast using random transmission power levels, our solution relies on the fact that the difference of RSS at the same location between two VPackets that are transmitted by the same RSU using two different power levels is roughly a constant across the entire communication

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

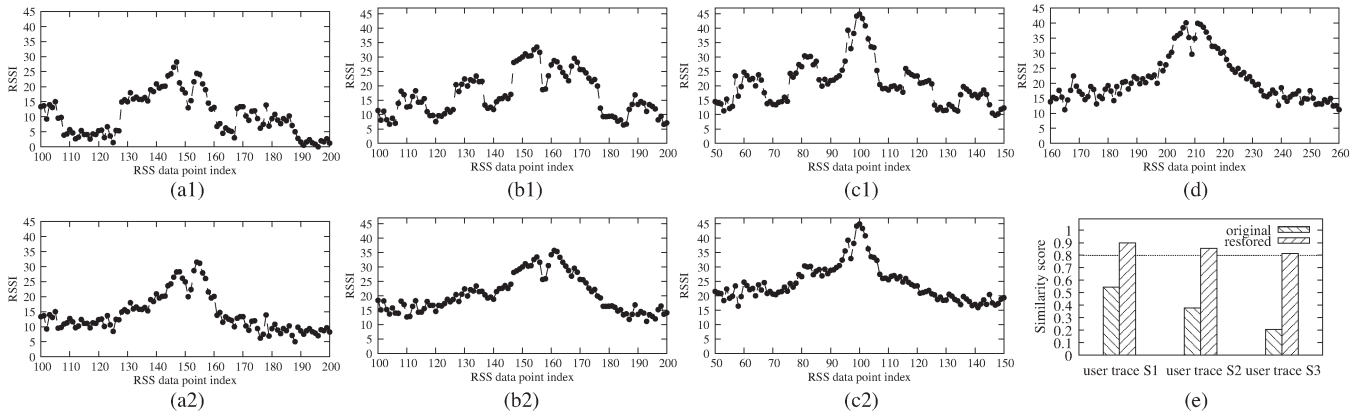



Fig. 4. RSS pattern comparisons of RSS series collected at road section $A$. (a1) Original RSS series of user trace S1; (a2) restored RSS series of user trace S1. (b1) Original RSS series of user trace S2; (b2) restored RSS series of user trace S2. (c1) Original RSS series of user trace S3; (c2) restored RSS series of user trace S3. (d) RSS series of the corresponding DB trace. (e) Similarity scores of user traces versus DB trace.

range of the RSU. We conducted an experiment to quantify how stable this RSS difference is. In the experiment, we let the roadside wireless AP change between two transmission power levels periodically while broadcasting the VPackets. In the first half of a coherence time period, the VPackets were transmitted using full power, and in the second half, the VPacket transmission power was set to half the full power. The VPacket rate was 100 frames/s. We drove a car past the RSU and collected the VPackets. We processed the received VPackets by first averaging the RSS of consecutive VPackets with the same transmission power. Then, we calculated the RSS difference between two sets of VPackets that were transmitted with two transmission power levels within a period of coherence time. Finally, we found that the average RSS difference is 8.55 dBm with a standard deviation of 2.81 dBm. As we will show later, this RSS difference is already stable enough that allows us to keep the fidelity of RSS patterns during the pattern restoration process.

*3) RSS Pattern Comparisons:* We have collected over 50 user traces at the three experiment locations, which are denoted as road sections $A$, $B$, and $C$, respectively. When comparing these honest user traces to their corresponding DB traces, we used an alphabet of four numbers (i.e., $K = 4$) at the quantization step. Empirically, we found that the similarity scores between all the user traces and the corresponding DB traces are larger than 0.8. Thus, in our scheme, we set the threshold similarity score $s_{\text{thresh}}$ as 0.8.

We show the RSS pattern comparisons for traces collected at road section $A$. Fig. 4(a1), (b1), and (c1) shows the patterns of three original RSS series constructed from three user traces, $S1$, $S2$, and $S3$, that were collected at road section $A$. Fig. 4(a2), (b2), and (c2) shows the patterns of the corresponding restored RSS series. Fig. 4(d) shows the pattern of the RSS series constructed from the DB's trace. We have only shown the center part of the patterns for better illustration. The randomly selected transmission power of VPackets allows the RSS of VPackets received within a short range of distance exhibit no fixed pattern. Therefore, we can see that the three original RSS series are all different (the similarity scores of any pair of the three original RSS series are less than 0.58). However, once we restore the original RSS series to their corresponding full-power RSS series, the three user traces do exhibit similar RSS patterns with the DB trace. Fig. 4(e) shows the similarity scores when comparing the DB trace against the original RSS series and against the three restored RSS series. It shows that the similarity scores between each original RSS series and the DB's RSS series are smaller than 0.55, and the similarity scores between each restored RSS series and the DB's RSS series are all larger than 0.81.

Another point worth mentioning is that of the first two user traces, i.e., $S1$ and $S2$, and the DB trace were collected using the Ubiquiti wireless card, whereas the third user trace $S3$ was collected using the Wistron wireless card. Although the RSS readings from the Wistron card were higher, it did not prevent our solution from correctly accepting the honest user trace, since our similarity comparison scheme only depends on the pattern itself but not on RSS reading amplitude.

*4) More Evaluation:* We have also performed the following experiments in our evaluation.

- We evaluated how VProof can deal with VPacket losses.
- We evaluated how VProof can defend against the case that a malicious user submits location proofs with guessed RSS values.
- We evaluated how the quantization alphabet size $K$ affects the similarity score of two RSS series.
- We evaluated the performance of different similarity comparison methods.

Due the page limit, the experiment details are presented in the full version of [10].

## VII. Conclusion

We have proposed VProof, which is a lightweight and privacy-preserving location proof solution that does not rely on PKI systems. We built a VProof prototype system and evaluated it with extensive experiments performed on actual road conditions. The evaluation results show that VProof can effectively verify if users' location claims match their historic locations without harming their location privacy.

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.

## ACKNOWLEDGMENT

The authors would like to thank all the reviewers for their insightful comments.

## REFERENCES

[1] IntelliDrive, Data Capture and Management Program, US DOT. [Online]. Available: www.its.dot.gov/research_docs/pdf/25Data%20Capture%20Federal%20Role.pdf
[2] CVIS: Cooperative Vehicle-Infrastructure Systems. [Online]. Available: www.cvisproject.org
[3] R. A. Popa, H. Balakrishnan, and A. J. Blumberg, "VPriv: Protecting privacy in location-based vehicular services," in *Proc. USENIX Security*, Berkeley, CA, USA, Aug. 2009, pp. 335–350.
[4] Y. Xi, K. Sha, W. Shi, L. Schwiebert, and T. Zhang, "Enforcing privacy using symmetric random key-set in vehicular networks," in *Proc. ISADS*, Mar. 2007, pp. 344–351.
[5] G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Lioy, "Efficient and robust pseudonymous authentication in VANET," in *Proc. VANET*, 2007, pp. 19–28.
[6] R. Lu, X. Lin, H. Zhu, P.-H. Ho, and X. Shen, "ECPP: Efficient conditional privacy preservation protocol for secure vehicular communications," in *Proc. INFOCOM*, Apr. 2008, pp. 1229–1237.
[7] V. Lenders, E. Koukoumidis, P. Zhang, and M. Martonosi, "Location-based trust for mobile user-generated content: Applications, challenges and implementations," in *Proc. HotMobile*, 2008, pp. 60–64.
[8] S. Saroiu and A. Wolman, "Enabling new mobile applications with location proofs," in *Proc. HotMobile*, 2009, pp. 3:1–3:6.
[9] W. Luo and U. Hengartner, "Proving your location without giving up your privacy," in *Proc. HotMobile*, 2010, pp. 7–12.
[10] Y. Zhang, C. C. Tan, F. Xu, H. Han, and Q. Li, "Lightweight and Privacy-Preserving Location Proofs for Intelligent Transportation Systems," College of William & Mary Department of Computer Science, Williamsburg, VA, USA, Tech. Rep., WM-CS-2014-05, 2014.
[11] K. Sampigethaya *et al.*, "CARAVAN: Providing location privacy for VANET," in *Proc. SCAR*, 2005, pp. 1–15.
[12] J. Freudiger, M. Raya, M. Flegyhzi, P. Papadimitratos, and J.-P. Hubaux, "Mix-Zones for location privacy in vehicular networks," in *Proc. WiN-ITS*, Aug. 2007, pp. 1–7.
[13] A. Wasef and X. Shen, "REP: Location privacy for VANETs using random encryption periods," *Mobile Netw. Appl.*, vol. 15, no. 1, pp. 172–185, Feb. 2010.
[14] Y. Zhang, Z. Li, and W. Trappe, "Power-modulated challenge-response schemes for verifying location claims," in *Proc. GLOBECOM*, 2007, pp. 39–43.
[15] X. Zheng *et al.*, "Characterizing the impact of multi-frequency and multi-power on localization accuracy," in *Proc. MASS*, Nov. 2010, pp. 156–156.
[16] H. Han, F. Xu, C. C. Tan, Y. Zhang, and Q. Li, "Defending against vehicular rogue APs," in *Proc. INFOCOM*, Apr. 2011, pp. 1665–1673.
[17] H. Han, F. Xu, C. C. Tan, Y. Zhang, and Q. Li, "VR-Defender: Self-defense against vehicular rogue APs for drive-thru Internet," *IEEE Trans. Veh. Technol.*, Mar. 11, 2014. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6762935&sortType%3Dasc_p_Sequence%26filter%3DAND(p_IS_Number%3A4356907).
[18] U.S. Department of Transportation, *IntelliDrive Data Capture and Management Program: Transforming the Federal Role*, 2010.
[19] C. Manasseh and R. Sengupta, "Middleware for Cooperative Vehicle-Infrastructure Systems," Partners for Advanced Transit and Highways, Richmond, CA, USA, Res. Rep. UCB-ITS-PRR-2008-2, 2008.
[20] P. Hu, B. Boundy, T. Truett, E. Chang, and S. Gordon, *Cross-Cutting Studies and State-of-the-Practice Reviews: Archive and Use of ITS-Generated Data*, 2002.
[21] B. Ostermaier, F. Dotzer, and M. Strassberger, "Enhancing the security of local danger warnings in VANETs—A simulative analysis of voting schemes," in *Proc. ARES*, Apr. 2007, pp. 422–431.
[22] M. Raya, P. Papadimitratos, V. D. Gligor, and J.-P. Hubaux, "On data-centric trust establishment in ephemeral ad hoc networks," in *Proc. IEEE Infocom*, Apr. 2008, pp. 1–11.
[23] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proc. KDD Workshop*, Apr. 1994, pp. 359–370.
[24] E. J. Keogh, "Exact indexing of dynamic time warping," in *VLDB*, 2002, pp. 1–12.
[25] J. Camp and E. W. Knightly, "Modulation rate adaptation in urban and vehicular environments: Cross-layer implementation and experimental evaluation," in *Proc. Mobicom*, 2008, pp. 315–326.

**Yifan Zhang** received the B.S. degree in computer science from Beihang University, Beijing, China, in 2004. He is currently working toward the Ph.D. degree in computer science with the College of William and Mary, Williamsburg, VA, USA.

His research interests include wireless networks, mobile computing systems, and operating systems.

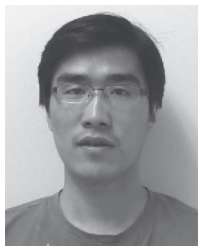
**Chiu C. Tan** (M'07) received the Ph.D. degree in computer science from the College of William and Mary, Williamsburg, VA, USA, in 2010.

He is an Assistant Professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA. His research is in the area of cyber security, and his current research interests include cloud computing security, smarthealth systems, and wireless network security (mainly 802.11, radio frequency identification, and sensor networks).

Dr. Tan is the Director for the National Science Foundation/Department of Defense Research Experience for Undergraduates Site program at Temple University.

**Fengyuan Xu** (M'13) received the Ph.D. degree in computer science from the College of William and Mary, Williamsburg, VA, USA, in 2013.

He is a Researcher with the Storage System Group, NEC Laboratories America. His research interests are in the broad areas of mobile and distributed systems, with a focus on smartphone energy efficiency, secure wireless communication, and data-intensive computing.



**Hao Han** received the Ph.D. degree in computer science from the College of William and Mary, Williamsburg, VA, USA, in 2013.

He is currently a Research Scientist with the Networks and Security Group, Intelligent Automation, Inc., Rockville, MD, USA. His research interests include wireless networks, mobile computing, cloud computing, and radio frequency identification systems.

**Qun Li** (SM'14) received the Ph.D. degree in computer science from Dartmouth College, Hanover, NH, USA, in 2004.

He is an Associate Professor with the Department of Computer Science, College of William and Mary, Williamsburg, VA, USA. His research interests include wireless networks, sensor networks, radio frequency identification, and pervasive computing systems.

Dr. Li received the National Science Foundation Career Award in 2008.

Authorized licensed use limited to: NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS. Downloaded on May 17,2023 at 11:49:03 UTC from IEEE Xplore. Restrictions apply.