Transparent and Accountable Vehicular Local Advertising With Practical Blockchain Designs

Dongxiao Liu, *Member, IEEE*, Jianbing Ni, *Member, IEEE*, Xiaodong Lin, *Fellow, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Vehicular local advertising enables roadside retailers to provide timely and location-aware advertisements to on-road vehicular users to attract more in-store visits. However, the prevalence of the vehicular local advertising has raised increasing transparency and accountability concerns on the spatial keyword query process, such as why a vehicular user receives advertisements from specific spatial entities as well as the timely detection of the advertising misbehavior. In this paper, we develop a transparent and accountable vehicular local advertising system by utilizing the tamper-proof and open nature of the blockchain technology. Considering the large scale of the spatial-keyword database and the expensive computation and storage cost on the blockchain, we introduce two design strategies, *digest-and-verify* and *divide-then-assemble*. In specific, a large-scale spatial keyword database is digested and stored on the blockchain. The spatial keyword query is then conducted with modular executions of two off-chain spatial keyword query functions, the results of which are efficiently assembled and verified in an advertising smart contract. By doing so, expensive on-chain computation and storage overheads are significantly reduced at a cost of acceptable off-chain overheads. We define the security requirements of the vehicular local advertising system as *Auditing Security* and achieve the notion with the security analysis. We also conduct real-world experiments to show the efficiency of the blockchain-based vehicular local advertising system.**

***Index Terms*—Vehicular local advertising, blockchain, smart contract, spatial keyword.**

I. Introduction

VEHICULAR local advertising is a prevalent advertising strategy, that allows roadside retailers (e.g. restaurants or retail stores) to promote targeted advertisements (ads) to nearby vehicular users of related interests. Vehicular users with Global Positioning System (GPS)-enabled devices (e.g. mobile phones) can receive timely and location-aware ads [1] through various vehicular communication channels [2], [3]. For example, vehicular users can find restaurants near their current locations, or receive coupons and flyers when they drive in a shopping center, which greatly increases their travel efficiency and experience. At the same time, vehicular local advertising also boosts in-store visits and promotes business for roadside retailers. According to statistics [4], an average of 80 percent of local searches convert and 28 percent of local searches lead to an in-store purchase. Therefore, vehicular local advertising is deemed as one of the most promising applications in intelligent transportation systems and has attracted extensive research and practice efforts.

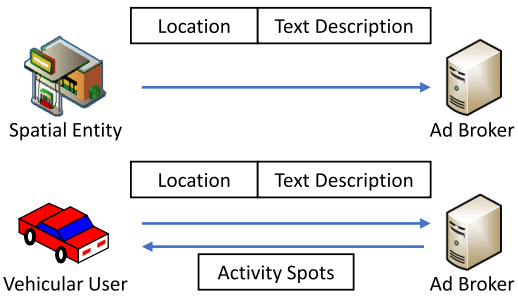


Fig. 1. Vehicular Local Advertising.

The ad dissemination process of the vehicular local advertising can be modeled as *spatial keyword query* [5]–[7], that considers both the spatial description (e.g., the retailer location and vehicle trajectory information) and textual description (e.g., retailer/user-specified keywords). In the vehicular local advertising, an advertising company can help retailers to conduct the spatial keyword query process. As shown in Fig. 1, retailers (spatial entities) can specify their locations and textual descriptions of their businesses. Spatial entities then update the location and textual descriptions to the advertising company (ad broker), that can send advertisements to vehicular users. Vehicular users can find nearby activity spots by sending a location with textual descriptions of their interests to the ad broker. However, the vehicular local advertising system has raised concerns and controversies recently on the lack of system *transparency* and *accountability* [8], [9]. *Transparency* requires the public visibility of the spatial keyword query process, which answers why a vehicular user receives the advertisements of specific spatial entities. It could result in the increasing popularity of ad blockers [9] at the vehicular users if without sufficient transparency guarantees. *Accountability* refers to two progressive meanings [10]. The first is the public detection of any breach of spatial keyword query protocols, such as the detection of *ad-fraud* attacks [11]. The second is the enforcement of obligations on the misbehaving

Manuscript received November 18, 2019; revised March 22, 2020 and July 28, 2020; accepted September 20, 2020. Date of publication October 20, 2020; date of current version January 22, 2021. The review of this article was coordinated by Prof. R. Qingyang Hu. *(Corresponding author: Xiaodong Lin.)*

Dongxiao Liu and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: dongxiao.liu@uwaterloo.ca; sshen@uwaterloo.ca).

Jianbing Ni is with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: jianbing.ni@queensu.ca).

Xiaodong Lin is with the School of Computer Science, University of Guelph, Guelph, ON N1G 2W1, Canada (e-mail: xlin08@uoguelph.ca).

Digital Object Identifier 10.1109/TVT.2020.3032375

advertisers or the ad broker. For example, a spatial entity that sends ads with the discrimination or misinformation should be suspended from the advertising service.

While the ad broker is working towards the increase of system transparency and accountability, it is still not sufficient in the view of the public [8], [12]. The emerging blockchain technology [13] can serve as a public infrastructure to enhance the current vehicular local advertising system [14], [15]. Essentially, a (public) blockchain is a secure and distributed database, that is maintained by an open and peer-to-peer network. As a result, it is tempting to migrate the vehicular local advertising system onto the blockchain, such that the spatial keyword query process can be open and verifiable to vehicular users. However, the design and implementation of a blockchain-based vehicular local advertising system faces non-trivial obstacles in terms of the execution practicality of the spatial keyword query. First, the large number of spatial objects with location and keyword information will result in a large-scale spatial keyword database. Second, the on-chain implementation cost in terms of the storage and computing is prohibitively expensive, since all blockchain nodes must maintain a local copy and verify the correctness of each transaction. As a result, a strawman solution that stores the large-scale spatial keyword database and executes the spatial keyword query on the blockchain is at the doubt of real-world practices. This motivates the main objective of this work: to take advantage of the principles of the blockchain technology for solving a real-world application with the practical designs. In particular, we aim at *practical* designs and implementations of a vehicular local advertising system that realizes the *transparency* and *accountability* (public detection of advertising misbehavior) promises of the blockchain technology.

In this paper, we exploit the tamper-proof and open nature of the blockchain technology and design a blockchain-based Transparent and Accountable Vehicular Local Advertising system (*TAVLA*). From the state-of-the-art building blocks, including the verifiable computation (VC) and spatial indexing techniques, *TAVLA* achieves the verifiable spatial keyword query process for the vehicular local advertising. *Verifiable* means that the spatial keyword query process is transparent to the public and any breach of spatial keyword query protocols can be effectively detected. Moreover, we address the scalability and efficiency issue of the blockchain-based advertising system with two design strategies, *digest-and-verify* and *divide-then-assemble*. The contributions of this paper are as follows:

- We develop a *verifiable* spatial keyword query scheme with a cryptographic query index, *SKD*-tree, that prunes the spatial keyword search space. Moreover, we construct a *TAVLA* smart contract, that realizes the *SKD*-tree on Ethereum and ensures sufficient public transparency and accountability of the spatial keyword query process.
- We introduce two design strategies for a practical blockchain-based vehicular local advertising system: *digest-and-verify* and *divide-then-assemble*. The large-scale spatial keyword database is digested and updated onto the blockchain with succinct cryptographic authenticators. The verifiable spatial keyword query scheme is then realized via modular executions of two off-chain verifiable functions. The results are assembled and verified in the *TAVLA* smart contract using the authenticators with limited on-chain storage and computation overheads.
- We conduct thorough security analysis to demonstrate that *TAVLA* achieves *Auditing Security* in terms of integrate, correct, transparent and accountable spatial keyword queries. We conduct experiments with the Pinocchio VC framework, which demonstrates that *TAVLA* is practical for implementations.

The organization of this paper is as follows. In Section II, we include the related work. In Section III, we revisit the state-of-the-art spatial keyword query techniques. Following definitions in Section III, we formulate the system model, security model and design goals of *TAVLA*. In Section V, we summarize the building blocks of *TAVLA*, including spatial indexing, verifiable computation, and smart contract techniques. In Section VI, we propose *TAVLA* in terms of an off-chain spatial keyword query scheme and an on-chain advertising contract. We present the security analysis and performance evaluation in Section VII and VIII, respectively. We conclude this paper in Section IX.

II. Related Work

Extensive research efforts have been directed to study the transparency in the location-based advertising systems. A comprehensive evaluation of the Facebook adverting system was conducted in [8]. The results demonstrated the lack of system transparency in the advertising system and the need of a better transparency auditing mechanism for the social network advertising system. Gui *et al.* [9] collected and analyzed a large volume of ad complaints from advertising users. Their results also revealed the users' lack of confidence on the contents of the received ads, which could lead to the installation of ad blocking softwares. Chen *et al.* [11] conducted extensive analysis of the existing mobile apps and ads. Their results showed the prevalence of fraud or irrelevant ads and an urgent need for a more transparent and accountable adverting system.

We briefly review the research line of *QAP*-based verifiable computations for general NP-complete relations. Gennaro *et al.* [16] proposed a framework that can compile NP-complete relations to a *QAP*, whose divisibility check is realized by linear combination checks in bilinear groups. A compiler that can convert a subset of C code into *QAP*s is later designed in [17]. Sequential work [18] made efforts on optimizing arithmetic circuit execution logics for loops and specialized relations. Fiore *et al.* [19] observed that a multi exponentiation (extended Pedersen commitment) of function inputs in the proof could be used multiple times. Modular design strategies for versatile statements are proposed in [20], [21], which achieve verifications on committed inputs and outputs. Eberhardt *et al.* [22] proposed a framework with programmable interfaces for outsourced verifiable computations. Bowe *et al.* [23] further formalized an on/off-chain computation model for decentralized verifiable computations.

We summarize the existing research works on building blockchian-base auditing infrastructures for public services. Li *et al.* [24] proposed a blockchain-based carpooling system,

TABLE I
AN ILLUSTRATIVE EXAMPLE OF TOPIC DESCRIPTIONS

	Game	Gym	...	Movie	Shop
T_1	0.67	0.03	...	0.03	0.03
T_2	0.03	0.03	...	0.91	0.03
...	...	...	...	...	...
T_{50}	0.04	0.04	...	0.75	0.04

which boosted the information exchange among carpooling companies and ensured the carpooling user privacy. An accountable geo-marketplace is proposed in [25], where spatial data can be privately queried based on geo-tags with the adoptions of symmetric searchable encryption techniques. Liu *et al.* [15] proposed a blockchain-based anonymous reputation management system, which increased the transparency of the reputation accumulation process. A blockchain-assisted ad dissemination scheme is proposed in [14], which motivated the vehicles to act as ad brokers with fair rewards.

TAVLA addresses transparency and accountability issues in the vehicular local advertising, by utilizing technical advances of verifiable computations and on/off chain models of the blockchain technology. However, considering the uniqueness of spatial keyword databases and the limitations of existing VC techniques, we non-trivially tailor the designs of *TAVLA* with practical implementations.

III. PRELIMINARIES

In this section, we revisit the state-of-the-art *non-verifiable* spatial keyword query technique. First, we introduce the vector space-based probabilistic topic model (**Definition 1**). Second, we formalize the definitions of spatial objects (**Definition 2**) and spatial keyword queries (**Definition 3**).

Definition 1: Probabilistic topic model [26] is a natural language processing mechanism that translates a textual description W of a spatial object o to a topic description T. For example, a textual description for a restaurant can be 'sea food', 'restaurant' and 'hot beverages'. A topic description is a collection of the topic distributions of W. In particular, T is an m-dimension vector. Each item $T[z]$ represents a relevance score of a topic z (intended activity) from an m-dimension topic dictionary D_T.

Latent Dirichlet Allocation (LDA) model [27] is adopted to translate the textual description W to a topic description T in the following equation:

$$T[z] = \frac{N_w + \epsilon}{|W| + |D_T| \times \epsilon} \tag{1}$$

N_w represents the number of keywords in W that belong to the topic z. $|W|, |D_T|$ is the size of the keyword, topic dictionary. ϵ is the symmetric Dirichlet prior [6]. In Table I, we show an illustrative example for topic descriptions of 50 spatial entities. Topics from D_T include 'Game', 'Gym', et al.. We can see that a spatial entity can relate to multiple topics (activities). A higher $T[z]$ indicates a larger relevance score of the intended activity z.

To measure the distance between two topic descriptions T and T', we adopt the Euclidean distance metrics as follows:

$$Dist(T, T') = \sqrt{\sum_{z=1}^{m} (T[z] - T'[z])^2} \tag{2}$$

Definition 2: A spatial object [6], [7] o_i for a spatial entity E_i is formally represented as:

$$o_i = (L_i, T_i)$$

$L_i = (x_i, y_i)$ is a two-dimension coordinate representation. T_i is the topic description of E_i.

Definition 3: A spatial keyword query function $\mathcal{F}_S$ takes into $\mathcal{O} = (o_1, o_2, ..., o_n)$ and a spatial keyword query $Q = (L_q, T_q, k, \lambda)$. It outputs top-$k$ spatial objects R_T from $\mathcal{O}$ that have the most higher relevance scores to the spatial keyword query Q.

$$\mathcal{F}_S(\mathcal{O}, Q) \rightarrow R_T$$

$L_q = (x_q, y_q)$ is a location of interest. T_q is the topic description of intended activities. λ is a preference ratio indicates the importance of the spatial proximity, while $1 - \lambda$ indicates the importance of the topic proximity. k is the number of spatial entities that will be returned. We define the point distance $Dist(L, L')$ with the Euclidean distance:

$$Dist(L, L') = \sqrt{(Lx - L'x)^2 + (Ly - L'y)^2} \tag{3}$$

Finally, a relevance score RS between a query Q and a spatial object $o_i = (L_i, T_i)$ is defined in Eq. 4 [6], [7], where η is the normalization factor to convert the point distance to a range of (0, 1).

$$RS(Q, o_i) = \lambda \times Dist(L_q, L_i)/\eta + (1 - \lambda) \times Dist(T_q, T_i) \tag{4}$$

IV. PROBLEM FORMULATION

In this section, we formulate the system model, security model, and design goals of *TAVLA*.

A. System Model

The system model of *TAVLA* is derived from the commercial model of the vehicular local advertising, such as local recommendation services in the Google map. The main difference is that *TAVLA* introduces the blockchain as a public auditing infrastructure to improve the transparency and enforce accountability for the advertising system. In Fig. 2, we abstract four parties in *TAVLA* as spatial entity, vehicular user, ad broker, and the blockchain.

- A spatial entity is a retailer on a road network, such as a restaurant or a coffee store. The spatial entity would like to promote their business by sending location-aware [28] ads to vehicular users of related interests.
- Vehicular users refer to moving vehicles on the roads with GPS-based devices. They would like to find spatial entities within a specific geographical area for various activities such as food, movie, gas and shopping.

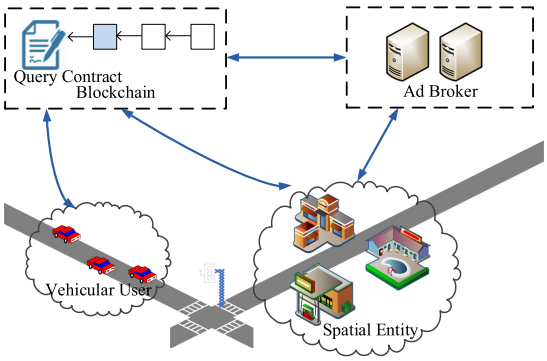


Fig. 2. System Model.

- The ad broker [29] provides third-party ad dissemination services for spatial entities. It is responsible for running spatial keyword queries and disseminating location-aware ads to targeted vehicular users.
- The blockchain in *TAVLA* is a public and appended-only ledger, which supports secure and verifiable transactions among mutually distrustful peer nodes. A smart contract can be deployed on the blockchain.

In *TAVLA*, spatial entities carefully specify spatial objects with their locations and topic descriptions, and send the objects to the ad broker. The ad broker constructs a spatial keyword database and uploads the cryptographic authenticators of the database onto the blockchain. Vehicular users construct the spatial keyword query Q and send it to an advertising contract on the blockchain. The ad broker retrieves spatial keyword queries from the contract, executes the queries over its local spatial keyword database, and sends back results (most relevant spatial entities) to the contract with correctness proof. The advertising contract verifies the correctness of the results and notifies the public if any verification fails. Finally, vehicular users retrieve the correct results from the advertising contract.

In *TAVLA*, the advertising smart contract is running on the Ethereum blockchain. It is critical to motivate the Ethereum miners to join the local advertising process. In the local advertising system, spatial entities will pay the ad broker for managing keywords and delivering ads to related vehicular users, which can be partially directed to the blockchain miners as 'reward' for managing the advertising smart contract.

B. Security Model

The spatial entities and ad broker are profit-driven commercial organizations: (1) Spatial entities carefully choose their locations and topic descriptions to enjoy effective ad dissemination services and attract more in-store visits. (2) Due to lack of advertising transparency and profit consideration, the ad broker may deviate from the spatial keyword query function in **Definition 3** and promote irrelevant or misleading ads to vehicular users [8]. Vehicular users accept the ad dissemination results if they are correctly generated. However, vehicular users may reject the results if there is lack of advertising transparency guarantees. Under the assumptions, we formulate the following *Auditing Security*:

Definition 4: Given a spatial keyword query function $\mathcal{F}_S$, that takes spatial objects $\mathcal{O}$ and a spatial keyword query Q as inputs and outputs a query result R_T, the *Auditing Security* has the following properties:

- *Integrity:* (1) Any $o_i \in \mathcal{O}$ digested in the on-chain authenticators cannot be maliciously modified in query executions. (2) Any Q generated by a vehicular user cannot be maliciously modified in query executions.
- *Correctness:* (1) R_T is a correct output of $\mathcal{F}_S$ with inputs $\mathcal{O}$ and Q. (2) Vehicular users accept the correct results.
- *Transparency:* The targeting keywords of spatial entities and the spatial keyword query process is transparent and verifiable to the public.
- *Accountability:* The misbehavior of the ad broker is publicly detected if the ad broker breaks either the integrity or correctness properties.

Remark: (1) The proposed blockchain-based architecture increases public transparency of the advertising system, which should contribute to the detection of advertising misconducts. It may also serve as the digital forensic evidence for law enforcement agencies to take actions against misbehaving parties. Moreover, the advertising smart contract can also take deposits from spatial entities and the ad broker and transfer the deposited money to vehicular users in case of any misconducts. (2) *TAVLA* relies on the pseudonym-based anonymity in the blockchain networks to protect vehicular users' query privacy. In specific, vehicular users interact with the advertising contract using anonymous blockchain accounts.

C. Design Goals

TAVLA aims at realizing the following design goals:

- Functionality: *TAVLA* should support the expressiveness of the spatial keyword query function in **Definitions 1–3**.
- Security: *TAVLA* should achieve the *Auditing Security* in **Definition 4**.
- Efficiency: *TAVLA* should optimize both the on/off-blockchain computation and storage overheads for practical implementations.

V. Building Blocks

In this section, we present the building blocks of *TAVLA*, including the spatial indexing technique, a verifiable computation framework, and the smart contract.

A. Spatial Indexing

An R-tree [7], [30] is widely used for indexing multi-dimension spatial data. It enables efficiently location searching that returns spatial objects within a given geographical area. In specific, an R-tree T_R in *TAVLA* is a balanced tree that contains the following components:

- Bounding Rectangle: It is a non-leaf node in the R-tree and represents a geographical area. Each bounding rectangle has n_b children in the R-tree, that represent smaller geographical areas within their parent.
- Minimum Bounding Rectangle (MB): It is a minimum geographical splitting area, specified by a lower-left point

TABLE II
NOTATIONS I

$\mathbb{G}$	Multiplicative groups
E_i	Spatial entity
$O = (o_1, o_2, ..., o_n)$	Collection of n spatial objects
$o_i = \{L_i, T_i\}$	Spatial object o_i Location $L_i = (x_i, y_i)$ m-dimension topic description T_i
$Q = \{L_q, T_q, k, \lambda\}$	Query location $L_q = (x_q, y_q)$ Query topic description T_q Number of returned spatial objects k Preference ratio λ
T_R	Spatial R-tree
$MB = (L_l, L_r)$	Minimum bounding rectangle MB Lower-left point L_l Upper-right point L_r
n_l	Number of MBs in T_R
n_e	Number of spatial objects in an MB

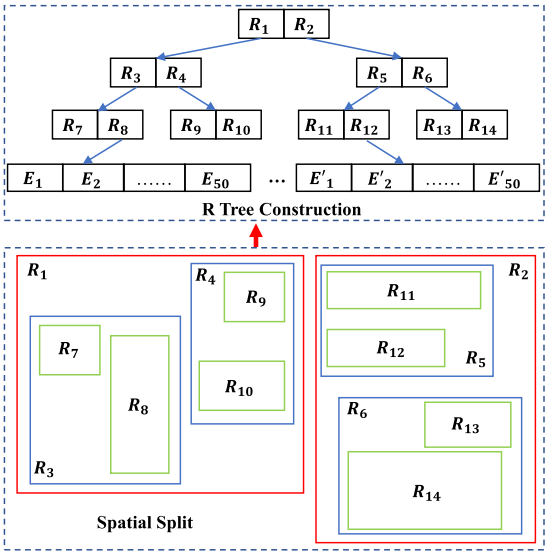


Fig. 3. An R-tree Example.

L_l, and a upper-right point L_r. Each MB consists of n_e spatial entities.

In Fig. 3 , the upper figure shows an R-tree example, while the lower figure represents a geographical area. The geographical area is first split into two subareas (bounding rectangles) (R_1, R_2), each of which is then split into smaller subareas (R_3, R_4, R_5, R_6). Recursively, the whole geographical area is split into $n_l = 8$ minimum bounding rectangles (R_7 to R_{14}). Each minimum bounding rectangle consists of $n_e = 50$ spatial entities. We define two algorithms of an R-tree as follows:

Definition 5: Given an R-tree T_R, it consists of two algorithms:

- **Search**. The algorithm takes into a location L and recursively queries T_R in a top-down manner to find an MB that contains L. The algorithm returns the identifier of the found MB.
- **Insert**. The algorithm takes into a spatial object o_i and inserts o_i into T_R. The algorithm finds an MB to place o_i using **Search** algorithm. If there is still room in MB, the algorithm inserts o_i into MB. Otherwise, the algorithm will *Split* MB into two smaller rectangles and *Adjust* T_R to remain balanced.

We omit the details of *Split* and *Adjust* algorithms. It is a lively research topic in the database area [5] and may be of independent research interests in the design of *TAVLA*.

B. Verifiable Computation

A VC framework [17] enables a verifier to outsource an execution of an NP-compete relation to a prover and verify the correctness of the execution results. In particular, a relation $\mathcal{R}$ is denoted as a public function $\mathcal{F}$. $\mathcal{F}$ takes into I, Q and outputs a result R. In a VC framework, $\mathcal{F}$ can be represented by a Quadratic Arithmetic Program (*QAP*) [16], which can be converted to the divisibility check of the *QAP* by a linear combination check with the Succinct Non-interactive ARGuments (*SNARG*) techniques. We present definitions of the Pinocchio VC framework [17] as follows:

Definition 6: There are three algorithms in VC framework:

$$VC = \{KeyGen,\ Evaluate, Verify\}.$$

- **KeyGen**($\mathcal{F}, pp$). The algorithm takes a function $\mathcal{F}$ and system public parameters pp. It outputs a relation-dependent common reference string $CRS = (ek, vk)$, where ek is for evaluations and vk is for verifications.
- **Evaluate**($\mathcal{F}, ek, I, Q$). The algorithm takes $\mathcal{F}, ek$ and function inputs I, Q. It outputs a function result R and a proof π_F.
- **Verify**(Q, R, π_F, vk). The algorithm takes the input Q, the result R, the proof π_F, and the verification key vk. It outputs *true* if $\mathcal{F}(I, Q) \rightarrow R$. Otherwise, it outputs *false*.

A VC framework should have the following properties:

- *Succinctness:* The length of the proof $|\pi_F|$ is polynomial in the system security parameter α.
- *Completeness:* An honest verifier will always accept (R, π_F), if $\mathcal{F}(I, Q) \rightarrow R$.
- *Soundness:* An computationally-bounded adversary cannot forge an invalid tuple (R', π'_F), where $\mathcal{F}(I, Q) \nrightarrow R$ and $Verify(Q, R', \pi'_F, vk) = true$.

C. Smart Contract

Smart contract [13], [31] is a computer program on the blockchain. It can take into crypto currencies as deposits from blockchain nodes, specify terms for the nodes, and take actions (transfer crypto currencies) if terms are met. In Ethereum, a smart contract is a special blockchain account with codes written in Solidity [32] on the blockchain. Participating blockchain nodes execute the smart contract by function calls as transactions to the smart contract address. In this way, each contract execution (state transition) is verified by all blockchain peer nodes. Based on the security and openness of the underlying blockchain, the Ethereum smart contract have two features: (1) Transparency: contract terms and executions are transparent to the public. (2) Security: contract executions are secure, which means the state transitions must follow the defined terms and cannot be later modified.

VI. Transparent and Accountable Vehicular Local Advertising

In this section, we present the blockchain-based *TAVLA*. First, we summarize the design strategies of *TAVLA*. Then, we design a verifiable spatial keyword query scheme with three phases: System Setup, *SKD*-tree Construction, and Spatial Keyword Query Processing. Finally, we develop a *TAVLA* smart contract that realizes the verifiable spatial keyword query scheme with on/off chain computation optimizations.

A. Overview

TAVLA utilizes the public blockchain as an auditing infrastructure, to achieve transparent and accountable ad dissemination process for the vehicular local advertising. To realize the promises of the blockchain technology with feasible storage and computation overheads, we introduce two design strategies as follows:

Digest-and-verify: The ad broker constructs query indexes of the spatial keyword database, digests the indexes as cryptographic authenticators, and uploads the authenticators onto to the blockchain. The on-chain authenticators have multiple functionalities. First, the authenticators are the evidence that the indexes are correctly digested. That is, spatial entities are able to check that their spatial objects are correctly digested in the indexes. Second, the ad broker can process spatial keyword queries in an off-chain manner. The correctness of the query results can be efficiently verified on the blockchain with the help of the cryptographic authenticators. With this strategy, the on-chain storage and computation overheads can be reduced to *succinct* regardless of the size of the spatial keyword database.

Divide-then-assemble: We identify that off-chain computation and storage overheads are dominated by the input size of a spatial keyword query function $\mathcal{F}_S$, that must be determined at the **KeyGen** phase in the VC framework. If $\mathcal{F}_S$ takes the whole spatial keyword database as inputs, it would incur prohibitive off-chain overheads. To enhance the off-chain performance, we adopt the probabilistic topic model to prune the keyword dimension of spatial objects. Meanwhile, we divide $\mathcal{F}_S$ into two verifiable functions: spatial search function $\mathcal{F}_R$ and topic matching function $\mathcal{F}_T$. $\mathcal{F}_R$ finds a minimum bounding rectangle MB_j that contains an intended location of a query Q. $\mathcal{F}_T$ finds top-k spatial objects with highest relevance scores in MB_j. The spatial keyword database is also divided into a spatial index and a topic index with distinct on-chain authenticators. As a result, $\mathcal{F}_S$ is realized with modular executions of $\mathcal{F}_R$ and $\mathcal{F}_T$, the results of which will be assembled on-chain with selective authenticators. With this strategy, *TAVLA* achieves a practical off-chain performance by reducing the input size of the off-chain functions.

B. Verifiable Spatial Keyword Query

In the following subsections, we present the details of the verifiable spatial keyword query scheme. We assume secure and authenticated channels are set up for involving entities. Notations in Definitions 1-6 are re-used.

TABLE III
Notations II

pp	Public Parameters
$\mathcal{F}_S = (\mathcal{F}_R, \mathcal{F}_T)$	Spatial keyword query $\mathcal{F}_S$ R-tree search $\mathcal{F}_R$, topic distance $\mathcal{F}_T$
CRS_R, CRS_T	CRS_R for $\mathcal{F}_R$ CRS_T for $\mathcal{F}_T$
$I = (I_R, I_T)$	Spatial index I_R Topic index $I_T = (I_1, ..., I_{n_l})$
$D = (D_R, D_T)$	Spatial index authenticator D_R Topic index authenticator D_T
$A[j]$	j-th element of an array A
π_R	Correctness proof for $\mathcal{F}_R$
π_T	Correctness proof for $\mathcal{F}_T$
π_D	Digest proof for D_R
$\hat{\pi}_D$	Digest proof for D_T
R_T	Spatial keyword query result

1) System Setup: For illustrative purposes, we introduce a trusted authority (TA) to setup the system. In practice, the role of TA can be replaced by a secure multi-party computation protocol [33]. Specifically, a set of entities (e.g. different ad brokers and randomly selected spatial entities, etc.) can agree on the spatial keyword query algorithm and setup the system. In practice, such setup mechanism has been successfully running for Zerocash system [34].

TA chooses a system security parameter α and sets the bilinear groups $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, e)$ with a prime order q and a bilinear paring e. TA chooses $g \in \mathbb{G}_1$ and $\tilde{g} \in \mathbb{G}_2$. TA denotes public parameters of the system as $pp = \{\alpha, \mathbb{G}, g, \tilde{g}\}$. TA divides the spatial keyword query function $\mathcal{F}_S$ as $(\mathcal{F}_R, \mathcal{F}_T)$ as follows:

$$\mathcal{F}_R(I_R, Q) \rightarrow MB_j, \mathcal{F}_T(I_j, Q) \rightarrow R_T \tag{5}$$

I_R is the spatial index of an R-tree T_R. Q is a spatial keyword query $Q = (L_q, T_q, k, \lambda)$. $\mathcal{F}_R$ finds a minimum bounding rectangle $MB_j \in T_R$ that contains L_q and outputs the identifier j of MB_j. I_j is the topic index of spatial objects in MB_j. $\mathcal{F}_T$ finds the top-k spatial objects in I_j that have highest relevance scores with Q and outputs identifiers of the k spatial objects as R_T. We note that algorithms of $\mathcal{F}_R$ and $\mathcal{F}_T$ are determined by TA at the setup phase, which will be discussed in details in *SKD*-tree Construction subsection.

TA computes common reference strings for $\mathcal{F}_R$ and $\mathcal{F}_T$. Since Pinocchio VC framework implements a non-updatable CRS model, TA must determine the size of I_R and I_j to generate the CRS. In specific, T_R is denoted as a balanced tree. n_b is the number of the bounding rectangles in each non-leaf node. n_l is the number of leaf nodes (MB). n_e is the number of spatial objects in each MB. TA runs $\mathbf{KeyGen}(\mathcal{F}_R, pp)$ and $\mathbf{KeyGen}(\mathcal{F}_T, pp)$ to generate $CRS = (CRS_R, CRS_T)$. TA randomly chooses $a, b, c \in \mathbb{Z}_q$ and computes $A = \tilde{g}^a, B = \tilde{g}^b, C = \tilde{g}^c \in \mathbb{G}_2^3$. Then, TA chooses $X = \{X_i\}_{i \in [1, 4n_l]}, Y = \{Y_i\}_{i \in [1, 4n_l]}$, where X_i, Y_i are randomly chosen from $\mathbb{G}_1$. Similarly, TA chooses $\hat{X} =$

Algorithm 1: Index Construction.

Input: $\mathcal{O} = (o_1, o_2, ..., o_n)$
Output: Spatial index I_R, topic index I_T
Set T_R, I_R, I_T to $\emptyset$
for $i \in [1, n]$ **do**
  Insert o_i into T_R
if *(number of MB in* T_R*)* $< n_l$ **then**
  Pack empty MBs to T_R
for $MB_i \in T_R$ **do**
  Add all $(L_l, L_r) \in MB_i$ to I_R
  for $o_j \in MB_i$ **do**
    Add $(L_j, T_j) \in o_j$ to I_i
  if *(number of objects in* I_i*)* $< n_e$ **then**
    Pack empty objects to I_i
  Add I_i to I_T

$\{\hat{X}_i\}_{i\in[1,(m+2)n_e]}, \hat{Y} = \{\hat{Y}_i\}_{i\in[1,(m+2)n_e]}$ from $\mathbb{G}_1$. TA computes $Z_i, \hat{Z}_j \in \mathbb{G}_1$ as follows:

$$\begin{aligned} Z_i &= X_i^a Y_i^b F_i^c,\ i \in [1, 4n_l],\ F_i \in CRS_R, \\ \hat{Z}_j &= \hat{X}_j^a \hat{Y}_j^b \hat{F}_j^c,\ j \in [1, (m+2)n_e],\ \hat{F}_i \in CRS_T. \end{aligned} \tag{6}$$

m is the dimension of the topic description T_i. $F = \{F_i\}_{i\in[1,4n_l]} \in \mathbb{G}_1^{4n_l}$ are from CRS_R. $\hat{F} = \{\hat{F}_i\}_{i\in[1,(m+2)n_e]} \in \mathbb{G}_1^{(m+2)n_e}$ are from CRS_T. TA sets $Z = \{Z_i\}_{i\in[1,4n_l]} \in \mathbb{G}_1^{4n_l}$, $\hat{Z} = \{\hat{Z}_i\}_{i\in[1,(m+2)n_e]} \in \mathbb{G}_1^{(m+2)n_e}$. TA denotes $K_D = (K_R, K_T, K_V)$, where $K_R = (F, X, Y, Z)$, $K_T = (\hat{F}, \hat{X}, \hat{Y}, \hat{Z})$, and $K_V = (A, B, C)$. TA publishes $\{pp, CRS, K_D\}$.

2) SKD-Tree Construction: Each spatial entity E_i constructs a spatial object $o_i = (L_i, T_i)$ and sends o_i to the ad broker. Upon receiving o_i, the ad broker will return a signature (e.g. ECDCS) on o_i as the proof of receipt. The ad broker collects all received spatial objects as a set $\mathcal{O} = (o_1, o_2, ..., o_n)$. Adopting algorithms in **Definition 5**, the ad broker constructs a spatial index I_R and topic index I_T in Algorithm 1.

Remark 5.1: (1) Our design is not coupled to a specific R-tree construction. As a result, *TAVLA* can naturally inherit technical advances (such as novel node split/deletion algorithms) for featured spatial databases. (2) Since the size of CRS is fixed in the setup phase, we pack empty MBs or spatial objects to I_R or I_i, to comply with the pre-determined index size n_l and n_e. An alternative strategy is that the ad broker first constructs the spatial database and requires an one-time CRS from TA.

We arrange I_R and I_T as arrays. I_R is a $4n_l$-dimension array, since each MB has four coordinates from (L_l, L_r). I_i is an $(m+2)n_e$-dimension array, since each $o_i \in I_i$ has an m-dimension topic description vector and a 2-dimension location L_i. The ad broker computes a spatial authenticator D_R and a topic authenticator $D_{T[i]}$ as follows:

$$\begin{aligned} D_R &= \prod X_i^{I_R[i]},\ \text{for } i \in [1, 4n_l], \\ D_{T[i]} &= \prod \hat{X}_j^{I_i[j]},\ \text{for } j \in [1, (m+2)n_e]. \end{aligned} \tag{7}$$

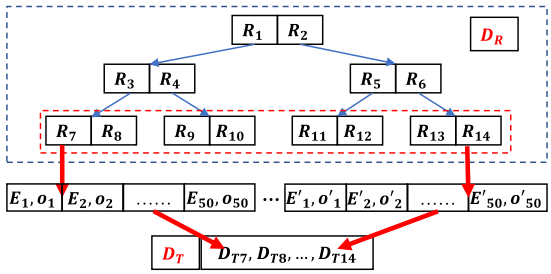


Fig. 4. An *SKD*-tree Example.

Algorithm 2: Query Execution.

Input: T_R, I_T, Q
Output: R_T
Run **Seach** with (T_R, L_q), find MB_j that contains L_q
for $o_i \in MB_j$ **do**
  Compute the relevance score $RS(Q, o_i)$
Add k objects with highest scores into R_T

$I_R[i]$ represents the i-th element in I_R and $I_i[j]$ represents the j-th element in I_j. The ad broker denotes $D_T = (D_{T[1]}, D_{T[2]}, ..., D_{T[n_l]})$ and uploads the *SKD*-tree authenticator $D = (D_R, D_T)$ to the TAVLA smart contract. A typical example is shown in Fig. 4. We can see that the *SKD*-tree is a balanced binary tree with 8 minimum bounding rectangles. I_R consists of MBs from R_7 to R_{14} with an authenticator D_R. Each MB contains 50 spatial objects. For example, R_7 contains spatial objects from o_1 to o_{50} with a topic authenticator $D_{T[7]}$.

Remark 5.2: We note that the keywords of spatial entities are managed by the ad broker in an off-chain manner and updated onto the blockchain with a succinct digest, which serve the following purposes. (1) Spatial entities can require the ad broker to 'open' the digest anytime and check if their spatial objects are correctly included in the authenticators. The ad broker cannot forge a false opening since he cannot solve the *Decision Diffie-Dellman* problem. (2) The ad broker can prove that each ad dissemination is correctly conducted following the ad dissemination protocol. By doing so, the digest becomes an immutable and auditable evidence of the advertising system.

3) Spatial Keyword Query Processing: Following **Definition 3**, we present modular designs of the spatial keyword query function, which consists of three algorithms:

$$\{QExe, QProv, QVeri\}.$$

QExe (Alg. 2) takes an R-tree T_R, a topic index I_T, and a query $Q = (L_q, T_q, k, \lambda)$. The algorithm outputs top k relevant spatial objects as R_T.

QProv (Alg. 3) takes the public parameters pp, $CRS = (CRS_R, CRS_T)$, the index (I_R, I_T), MB_j, the query Q, the authenticators (D_R, D_T), and the digest keys K_D. It outputs proofs $\pi = (\pi_R, \pi_T, \pi_D, \hat{\pi}_D)$.

QVeri (Alg. 4) verifies the correctness of the proofs.

Remark 5.3: (1) Modular uses of the building blocks are achieved in *TAVLA*, since the same building blocks are sufficiently abstracted in the previous section and are used multiple

Algorithm 3: Query Prove.

Input: $pp, CRS, (I_R, I_T), MB_j, Q, (D_R, D_T), K_D$
Output: π
Run $Evaluate(\mathcal{F}_R, CRS_R.ek, I_R, Q)$ to generate π_R
Run $Evaluate(\mathcal{F}_T, CRS_T.ek, I_j, Q)$ to generate π_T
Compute $Z_\pi = \prod_{i\in[1,4n_l]} Z_i^{I_R[i]}$
Compute $Y_\pi = \prod_{i\in[1,4n_l]} Y_i^{I_R[i]}$
Compute $\hat{Z}_\pi = \prod_{i\in[1,(m+2)n_e]} \hat{Z}_i^{I_j[i]}$
Compute $\hat{Y}_\pi = \prod_{i\in[1,(m+2)n_e]} \hat{Y}_i^{I_j[i]}$
Set $\pi_D = (Z_\pi, Y_\pi)$, $\hat{\pi}_D = (\hat{Z}_\pi, \hat{Y}_\pi)$
Set $\pi = (\pi_R, \pi_T, \pi_D, \hat{\pi}_D)$

Algorithm 4: Query Verify.

Input: $CRS, Q, D, K_V, (j, R_T), \pi$
OutPUT: True or false
Check $Verify(Q, j, \pi_R, CRS_R.vk)$
Check $Verify(Q, R_T, \pi_T, CRS_T.vk)$
Extract $c_x = \prod F_i^{I_R[i]}$, $i \in [1, 4n_l]$ from π_R
Extract $\hat{c}_x = \prod \hat{F}_i^{I_j[i]}$, $i \in [1, (m+2)n_e]$ from π_T
Check $e(Z_\pi, \tilde{g}) \stackrel{?}{=} e(D_R, A)e(Y_\pi, B)e(c_x, C)$
Check $e(\hat{Z}_\pi, \tilde{g}) \stackrel{?}{=} e(D_{T[j]}, A)e(\hat{Y}_\pi, B)e(\hat{c}_x, C)$
Return *true* if all checks pass. Otherwise, return *false*

times. (2) We re-design functions $\mathcal{F}_R$ and $\mathcal{F}_T$ to fit the *digest-and-verify* strategy, which are implemented in C codes and will be discussed in the performance evaluation section.

C. TAVLA *Smart Contract*

We design a *TAVLA* smart contract that realizes the verifiable spatial keyword query scheme based on the Ethereum [32]. The contract is created by TA, that stores public parameters pp, verification keys ek_R, ek_T, digest verification keys K_V, and digest authenticators $D = (D_R, D_T)$. The details of *TAVLA* contract is shown in Alg. 5. The definitions and algorithms proposed in the previous sections are re-used.

Vehicular users call *SendQuery* function to send spatial keyword queries to the contract, where $addr$ is the blockchain address of the message sender. The ad broker retrieves unprocessed queries using the *RetrieveQuery* function. The ad broker executes the queries locally via **QProv** and **QExe** function, and uploads the results with proofs to the smart contract via the *SendResult* function. The correctness of the result and proofs are verified by the smart contract. The smart contract stores valid results to be retrieved by vehicular users via *QueryResult* function. For invalid results, the smart contract generates a verification failure event to notify the public.

Remark 5.4: (1) Vehicular user privacy is not a primary concern of *TAVLA*. Vehicular users can apply for one-time blockchain accounts and utilize anonymous payment channels (such as Zerocash). (2) The main goal of the contract is to improve the advertising system transparency and accountability.

Algorithm 5: *TAVLA* Smart Contract.

Require: $pp, (ek_R, ek_T)$, K_V, $D = (D_R, D_T)$
Set Rec_Q to $(addr, Q, flag)$
Set Rec_R to $(addr, MB_j, R_T)$
Function `SendQuery` (*a spatial keyword query* Q)
  Set $addr = sender.addr$, $flag = 0$
  Add $(addr, Q, flag)$ to Rec_Q
Function `RetrieveQuery()`
  Require: msg.sender=ad broker
  Retrieve all $(addr, Q)$ from Rec_Q with $flag = 0$
Function `SendResult` $(addr, (j, R_T), \pi)$
  Require: msg.sender=ad broker
  Retrieve $(Q, flag)$ from Rec_Q by $addr$
  Require: $flag = 0$
  if ***QVeri***$(CRS, Q, D, K_V, (j, R_T), \pi) = true$ **then**
    Set $flag$ to 1
    Add $(addr, j, R_T)$ to Rec_R
  else
    Generate a verification failure event
Function `QueryResult()`
  Set $addr$ to msg.sender
  Retrieve $(addr, j, R_T)$ from Rec_R

The accountability in *TAVLA* refers to the *detection* or *public awareness* of the ad dissemination misbehavior. Potential obligations on the misbehavior can be transfers of the ad broker's pre-deposited crypto currencies or enforcing fines by law enforcement agencies.

VII. Security Analysis

In this section, we present the security analysis of *TAVLA*. We first review the security properties inherited from the cryptographic building blocks: the blockchain and the verifiable computation framework. Then, we demonstrate the security properties of *Auditing Security*.

A. Blockchain Security

The consensus protocol of a public blockchain (i.e. Proof-of-Work in Ethereum) provides three useful properties: *chain growth*, *chain quality* and *consistency* [13], [35]. Informally, the three properties guarantee: (1) a valid transaction will be accepted by honest blockchain nodes within a certain time (transaction confirmation time); (2) a Byzantine adversary that controls less than 50 percent computation power of the blockchain system cannot control the growth of the chain; and (3) honest blockchain nodes maintain a consistent view of the shared ledger.

B. Verifiable Computation Framework Security

Completeness: An honest verifier always accepts a result and a proof if they are correctly computed. For Pinocchio VC framework, the *QAP*-based *SNARG* system [16] recognizes an NP-complete relation that can be compiled to an arithmetic circuit C. From the *QAP* theorem, the circuit evaluation of C

can be conducted by checking the divisibility of the compiled polynomials. The *QAP* divisibility check is further converted to a linear check over bilinear groups. From *QAP* theorem and bilinear groups, the *completeness* is achieved.

Soundness: A computationally-bounded adversary (usually refers to a malicious prover) cannot forge an invalid result with a proof that passes the correctness check (*Verify* function in the VC framework). For a compiled *QAP* with a degree d and bilinear groups with an order q, we borrow the theorem from [16] that *Soundness* is achieved if (1) **q-PDH, 2q-SDH**, and **d-PKE** assumptions hold for $q \geq 4d + 4$. (2) The trapdoor secret used to generate common reference strings is destroyed.

Succinctness: The VC framework achieves a succinct size of proof that depends on the size of the system security parameter α regardless of the function input size.

C. TAVLA *Security*

Based on the above security properties from the building blocks, we give a sketch analysis of the security properties in **Definition 4**.

Integrity: Spatial objects are generated by individual spatial entities and sent to the ad broker in a secure channel. Then, the ad broker computes the authenticators D_R and D_T to be uploaded onto the blockchain. To ensure the individual spatial object is digested in the authenticator, the spatial entity can require the ad broker to compute a proof for the correct authenticator generation, which is either a zero-knowledge proof of a linear relation in the discrete logarithm setting or a direct opening of the targeting keywords. The query smart contract receives spatial keyword queries from vehicular users and verifies the correctness of query executions using the on-chain authenticators. If the on-chain storage and advertising contract executions are secure in the Ethereum blockchain, the *Integrity* property is achieved in *TAVLA*.

Correctness: The first property of *Correctness* comes from three folds: (1) *Soundness* of the underlying VC framework. (2) *Integrity* of spatial objects and spatial keyword queries. (3) Unforgeability of the on-chain authenticators. For each spatial keyword query, the ad broker performs the query over the spatial and topic index I_R and I_T. In specific, the ad broker runs the *Evaluate* function of $\mathcal{F}_R$ and $\mathcal{F}_T$ and proves the correctness of query executions. In *TAVLA*, CRS_R and CRS_T are securely generated by TA or a secure multiparty computation protocol. The ad broker can forge a query result with a proof that passes the *Verify* function of $\mathcal{F}_R$ and $\mathcal{F}_T$, iff the ad broker can break the *Soundness* property of the VC framework. The proof π_R and π_T contain multi-exponentiation forms c_x and $\hat{c}_x$ as the representations of I_R and I_T. In specific, c_x and $\hat{c}_x$ follow the form of extended Pedersen commitment for vectors. The ad broker proves that the c_x and $\hat{c}_x$ open to the same value of the on-chain authenticators (D_R, D_T). The ad broker can forge valid proofs $(\pi'_D, \hat{\pi}'_D)$ that pass the check in Alg. 4, iff the ad broker can solve the **SXDH** problem in bilinear groups [19]. For the second property of *Correctness*, honest vehicular users will always accept valid results and proofs due to the *Completeness* property of the VC framework.

Transparency: The Ethereum blockchain is a public and permissionless ledger. Since the succinct digest of the spatial objects are uploaded to the Ethereum, vehicular users can require the ad broker to publish the keywords of a specific spatial entity for further authenticity checking. At the same time, executions of the local advertising contract are also transparent to the public.

Accountability: We emphasize that *Accountability* of *TAVLA* refers to the public *detection* or *awareness* of the ad broker misbehavior. Spatial entities can require the ad broker to publish the proof of correct authenticator generation, while vehicular users directly receive the query results on the smart contract, both of which can be publicly verifiable.

VIII. Performance Evaluation

A. Off-Chain Overheads

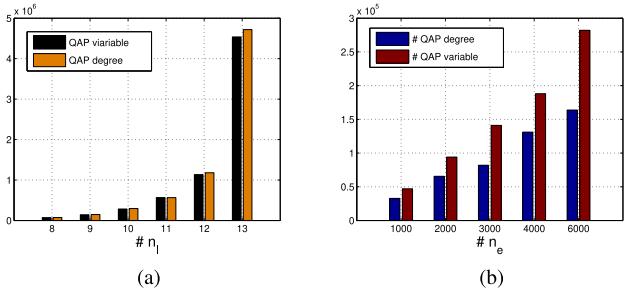
Off-chain operations include System Setup, *SKD*-tree Construction, and Spatial Keyword Query Processing. We omit implementations and evaluations of the R-tree and probabilistic topic model, which have been well studied in the non-verifiable setting. In *TAVLA*, our implementation goal is to evaluate additional overheads with the implementation of the VC framework. Thus, we construct testing instances of a balanced R-tree and topic descriptions. With the testing instances, we evaluate performances of $\mathcal{F}_R$ and $\mathcal{F}_T$, in terms of off-chain storage and computation overheads.

We conduct off-chain experiments on a Linux system with Intel Core 2.4 GHz processor and 8 GB memory. The functions $\mathcal{F}_R$ and $\mathcal{F}_T$ are written in C codes. We implement the Pinocchio [17] VC framework that translates the query execution into arithmetic circuits. We note that the Pinocchio interface is compiled with a 32-bit version gcc. We write a circuit parser in C++ to parse the obtained circuits with 'nizk' circuit inputs (both spatial query Q and spatial objects $\mathcal{O}$) and implement the C++ interface of libsanrk [18], [36], [37] for *R1CS* languages. Our design is not specified to particular implementations of *QAP*-based VC framework. Thus, *TAVLA* can inherit any efficiency improvements of future optimizations for *QAP*-based verifiable computations [18].

We have found several implementation limitations in the circuit-based VC framework: (1) The Pinocchio C program compiler only supports static compilation, which requires fixed-size spatial and topic indexes as inputs [38]–[40]. (2) Subscripts of the array access in the C codes must be determined at the program compiling phase. Thus, the logarithmic R-tree search algorithm cannot be implemented. Instead, we implement a linear search algorithm over the leaf nodes in the spatial index. In specific, we must compare L_q of Q with (L_l, L_r) of each MB. (3) The Pinocchio C program compiler only supports integers and simple arithmetic operations. We re-write the relevance score function in Eq. 4 as follows:

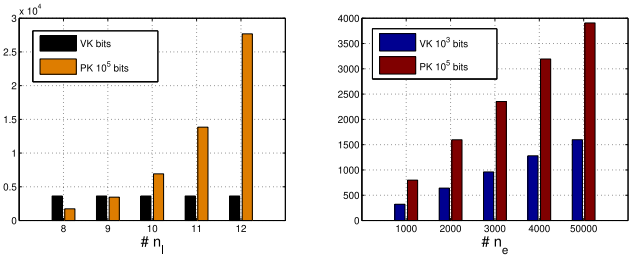
$$\lambda_1 \times Dist^2(L_q, L_i) + \lambda_2 \times Dist^2(T_q, T_i)$$

The location coordinates L_q, L_i, topic descriptions T_q, T_i and preference factors λ_1, λ_2 are set as integers in the experiments.



(a) (b)

Fig. 5. *QAP* Complexity. (a) $\mathcal{F}_R$. (b) $\mathcal{F}_T$.



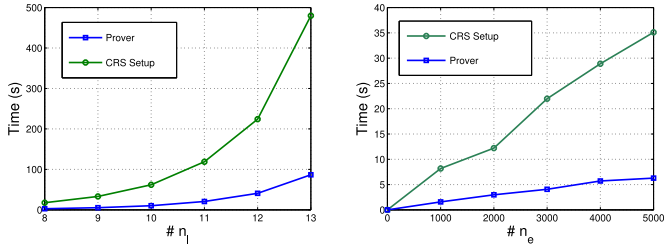
(a) (b)

Fig. 6. CRS Size. (a) $\mathcal{F}_R$. (b) $\mathcal{F}_T$.

Square root computations of point and topic distances are eliminated. However, the accuracy of the relevance ranking may be affected compared with the original metric in Eq. 4. Thus, we set a larger k in our experiments. Similarly, a local search in Google Map for a specific area will return all the relevant results to users.

We first identify the main off-chain performance metrics. In specific, the complexity of the compiled quadratic arithmetic program is characterized by the *QAP* variables and degrees. The storage overhead is characterized by the size of CRS. Another important metric is the off-chain processing time of the two functions, in terms of CRS setup and prover computation. We set the number of bounding rectangles n_b as 2, which results in a binary tree structure and is easy to be adjusted to balance when new nodes are inserted. We set the number of returned objects k to be equal to n_e. This is reasonable since a spatial keyword query usually returns all relevant result to users, such as activity spot search in Google Map. The dimension m of the topic description vector is 20, which is sufficient in the probabilistic topic model [6]. The normalization factor η can be adjusted with the change of λ_1. Input sizes of $\mathcal{F}_R$ and $\mathcal{F}_T$, characterized by (n_e, n_l), greatly affect the performance. n_l is represented as the power of 2, since the R-tree is a balanced binary tree in the experiments. From Fig. 5 a and 5 b, we can see that the *QAP* complexity is increasing with n_l and n_e with different rates.

In Fig. 6, the same increasing property with n_l and n_e is found for PK size. In Pinocchio vc framework, the VK size is determined by the number of plaintext inputs and outputs. Since we enable 'nizk' for all the input, VK size is solely determined by the number of outputs, which results in a constant-size VK in $\mathcal{F}_R$ and a linearly increasing size of VK in $\mathcal{F}_T$. PK size is much more larger (10^9 magnitude) compared with VK size, since PK must embed information of both input and intermediate gates in the compiled circuits, while VK only embeds information of output gates.



(a) (b)

Fig. 7. Off-chain Computation Cost. (a) $\mathcal{F}_R$. (b) $\mathcal{F}_T$.

TABLE IV
DIGEST COST VS n_l, n_e

	K_R, K_T Size	D_R, D_T Comp.	$\pi_D, \hat{\pi}_D$ Comp.
$\mathcal{F}_R$	$16n_l\|\mathbb{G}_1\|$	$4n_lE_1$	$8n_lE_1$
$\mathcal{F}_T$	$4(m+2)n_e\|\mathbb{G}_1\|$	$(m+2)n_eE_1$	$2(m+2)n_eE_1$

In Fig. 7, the prover overhead refers to the *Evaluate* algorithm in the VC framework. The one-time CRS setup is much costive. Meanwhile, the processing time is also increasing as n_l and n_e grow. It should be noted that the prover overhead with an input of a few thousand objects is a few seconds on a laptop, which can be improved at the ad broker with powerful computing clusters. Moreover, distributed and parallel optimization techniques for verifiable computations can also be adopted to further enhance the prover performance.

We summarize the storage and computing complexity of cryptographic authenticators in Table IV. Although the complexity increases greatly with the input size (n_l, n_e), it is still less significant than the CRS setup and prover computation overheads. The reason is that multi-exponentiation operation is extremely optimized in the *alt-bn128* and *bn-128* curves in the libff library [41] of *libsnark*. For example, a 254-bit multi-exponentiation operation only takes 231.2 μs on a 2.4 GHz Intel Xeon E5-2620 [19]. $|\mathbb{G}_1|$ refers to the size of an element in $\mathbb{G}_1$. E_1 is one exponentiation operation in $\mathbb{G}_1$. The *digest-and-verify* strategy may slightly increase the size of π_R and π_T in our off-chain experiments with the *libsnark* implementation by requiring independent components $c_x, \hat{c}_x$.

B. On-Chain Overheads

We implement a blockchain network based on Parity Ethereum [42]. The VC framework is instantiated with *alt-bn128* curve, which is compatible for the pre-compiled pairing interface in Ethereum [13]. Since *TAVLA* smart contract is designed for the Ethereum blockchain, we take the main performance metrics from the Ethereum to evaluate the *TAVLA* smart contract: on-chain storage cost, transaction confirmation time and gas cost of function calls. The 'gas' is a unit in the Ethereum to measure the computation complexity of a transaction.

Storage cost: The on-chain storage includes the verification keys for $\mathcal{F}_R$ and $\mathcal{F}_T$, the index authenticators, and the digest verification keys. VK size is determined by the number of non

zero-knowledge variables. The authenticator size for the spatial and topic index $I = (I_R, I_T)$ is $(n_l + 1)|\mathbb{G}_1|$. The size of the digest verification key $K_V = (A, B, C)$ is $3|\mathbb{G}_2|$.

Function call: In Ethereum, each function call is instantiated by an Ethereum transaction. The transaction confirmation time can be approximated by ETH status [43]. The most expensive function call is the *SendResult* function, which is dominated by the number of pairing operations in the function. The verifications of $\mathcal{F}_R$ and $\mathcal{F}_T$ include 13×2 pairings. Compared with our off-chain experiments with the *libsnark* implementation, *TAVLA* needs 1 more pairing for the verification, since the verifier needs to check that the appropriate span of c_x or $\hat{c}_x$. The authenticator checks require 4×2 pairings. According to statistics in EIP 1108, total verification cost is approximately 1,201,000 gas. Note that, the *RetrieveQuery* and *QueryResult* can be conducted without sending transactions to the smart contract. A node with a full copy of the blockchain storage can locally query the contract status.

C. On/off Chain Tradeoffs

The on-chain cost can be reduced to constant with one authenticator for the whole spatial and topic databases. However, the on-chain strategy will introduce infeasible off-chain cost, since off-chain cost is linearly increasing with the input size of $\mathcal{F}_R$ and $\mathcal{F}_T$. We identify two split factors that can quantify the on/off chain tradeoffs: f_r for the spatial database and f_t for the topic database. In *TAVLA*, f_r is set to 1 as $\mathcal{F}_R$ takes the whole spatial index as the input. f_t is set to n_l, as $\mathcal{F}_T$ takes into a subindex I_j of topic index I_R. For very large spatial databases, the spatial index can be split into different subindexes. The split factors serve as the tradeoff switches to tune the on/off chain performance: higher f_t and f_r increase the on-chain overheads and reduce off-chain overheads.

Discussion: To further increase the real-time processing capability [44] of the advertising system, we present measures for real-world implementations. (1) The ad broker can adopt a distributed *SNARGs* to implement $Evaluate$ function over the powerful computing clusters. (2) The ad broker can tune the on/off-chain tradeoff by dividing the whole spatial database into sub databases. For example, spatial objects can be organized around some hot spots. (3) It has been proven that the scalability of a blockchain system can be significantly improved with novel consensus protocols, such as Proof-of-Stake or Byzantine Fault Tolerant. We emphasize that our design strategies of *TAVLA* do not rely on specific blockchain architectures, and thus can be implemented with efficient consensus protocols.

IX. Conclusion

In this paper, we have proposed a blockchain-based transparent and accountable vehicular local advertising system. With the two design strategies, the proposed system has achieved significant on-chain computation and storage efficiency. Through extensive experiments, we have explored the implementation challenges for the vehicular local advertising system to provide comprehensive performance benchmarks. The practical design strategies and the observed implementation insights may shed light on future constructions of blockchain-based vehicular applications. In our future work, we will further investigate the privacy preservation of vehicular users and design a transparent yet privacy-preserving blockchain-based vehicular advertising system.

References

[1] K. Suto, H. Nishiyama, and N. Kato, "Postdisaster user location maneuvering method for improving the QoE guaranteed service time in energy harvesting small cell networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9410–9420, Oct. 2017.

[2] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10 190–10 203, Nov. 2018.

[3] F. Lyu *et al.*, "SS-MAC: A novel time slot-sharing MAC for safety messages broadcasting in vanets," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3586–3597, Apr. 2017.

[4] Local Search Statistics. 2019, [Online]. Available: https://www.webfx.com/blog/marketing/google-ads-statistics/, Accessed: Mar. 2020.

[5] Z. Li, L. Chen, and Y. Wang, "G*-tree: An efficient spatial index on road networks," in *Proc. IEEE 35th Int. Conf. Data Eng.*, 2019, pp. 268–279.

[6] Z. Qian, J. Xu, K. Zheng, P. Zhao, and X. Zhou, "Semantic-aware top-k spatial keyword queries," *World Wide Web*, vol. 21, no. 3, pp. 573–594, 2018.

[7] D. Zhang, Y. Li, X. Cao, J. Shao, and H. T. Shen, "Augmented keyword search on spatial entity databases," *VLDB J.*, vol. 27, no. 2, pp. 225–244, 2018.

[8] A. Andreou, G. Venkatadri, O. Goga, K. Gummadi, P. Loiseau, and A. Mislove, "Investigating ad transparency mechanisms in social media: A case study of facebook's explanations," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018. [Online]. Available: http://dx.doi.org/10.14722/ndss.2018.23191

[9] J. Gui, M. Nagappan, and W. G. Halfond, "What aspects of mobile ads do users care about? An empirical study of mobile in-app ad reviews," 2017, *arXiv:1702.07681*.

[10] J. Frankle, S. Park, D. Shaar, S. Goldwasser, and D. Weitzner, "Practical accountability of secret processes," in *Proc. USENIX Secur. Symp.*, 2018, pp. 657–674.

[11] G. Chen, W. Meng, and J. Copeland, "Revisiting mobile advertising threats with madlife," in *Proc. ACM World Wide Web Conf.*, 2019, pp. 207–217.

[12] J. Parra-Arnau, J. P. Achara, and C. Castelluccia, "Myadchoices: Bringing transparency and control to online advertising," *ACM Trans. Web*, vol. 11, no. 1, pp. 1–47, 2017.

[13] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Ethereum Project Yellow Paper, pp. 1–39, 2018-06-05.

[14] M. Li, J. Weng, A. Yang, J.-N. Liu, and X. Lin, "Toward blockchain-based fair and anonymous ad dissemination in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11 248–11 259, Nov. 2019.

[15] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for IIoT-enabled retail marketing atop PoS blockchain," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3527–3537, Jun. 2019.

[16] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, "Quadratic span programs and succinct NIZKs without PCPs," in *Proc. EUROCRYPT*. Springer, 2013, pp. 626–645.

[17] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *Proc. IEEE Symp. Secur. Privacy*, 2013, pp. 238–252.

[18] A. Kosba, C. Papamanthou, and E. Shi, "xJsnark: A framework for efficient verifiable computation," in *Proc. IEEE Symp. Secur. Privacy*, 2018, pp. 944–961.

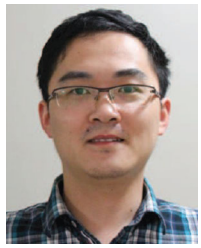
[19] D. Fiore, C. Fournet, E. Ghosh, M. Kohlweiss, O. Ohrimenko, and B. Parno, "Hash first, argue later: Adaptive verifiable computations on outsourced data," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2016, pp. 1304–1316.

[20] S. Agrawal, C. Ganesh, and P. Mohassel, "Non-interactive zero-knowledge proofs for composite statements," in *Proc. CRYPTO*, 2018, pp. 643–673.

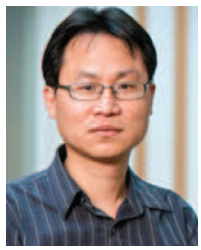
[21] M. Campanelli, D. Fiore, and A. Querol, "Legosnark: Modular design and composition of succinct zero-knowledge proofs," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2019, pp. 2075–2092.

[22] J. Eberhardt and S. Tai, "Zokrates-scalable privacy-preserving off-chain computations," in *Proc. IEEE Int. Conf. Blockchain*, 2018, pp. 1084–1091.
[23] S. Bowe, A. Chiesa, M. Green, I. Miers, P. Mishra, and H. Wu, "Zexe: Enabling decentralized private computation," in *Proc. IEEE Symp. Secur. Privacy*, 2020, pp. 947–964.
[24] M. Li, L. Zhu, and X. Lin, "Efficient and privacy-preserving carpooling using blockchain-assisted vehicular fog computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4573–4584, Jun. 2019.
[25] K. Nguyen, G. Ghinita, M. Naveed, and C. Shahabi, "A privacy-preserving, accountable and spam-resilient geo-marketplace," in *Proc. ACM SIGSPATIAL*, 2019, pp. 299–308.
[26] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou, "Short text understanding through lexical-semantic analysis," in *Proc. IEEE Int. Conf. Data Eng.*, 2015, pp. 495–506.
[27] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.
[28] N. Kato *et al.*, "Location awareness system for drones flying beyond visual line of sight exploiting the 400 MHz frequency band," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 149–155, Dec. 2019.
[29] Google Ads. [Online]. Available: https://support.google.com/adwords/, Accessed: Oct. 2019.
[30] A. Guttman, "R-trees: A dynamic index structure for spatial searching," *ACM SIGMOD Rec.*, vol. 14, no. 2, pp. 47–57, 1984.
[31] S. Hu, C. Cai, Q. Wang, C. Wang, X. Luo, and K. Ren, "Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization," in *Proc. IEEE INFOCOM*, 2018, pp. 792–800.
[32] Solidity. [Online]. Available: https://solidity.readthedocs.io/en/v0.4.25/, Accessed: Oct. 2018.
[33] S. Bowe, A. Gabizon, and M. D. Green, "A multi-party protocol for constructing the public parameters of the pinocchio zk-SNARK," in *Proc. FC*, 2019, pp. 64–77.
[34] E. B. Sasson *et al.*, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, 2014, pp. 459–474.
[35] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," in *Proc. EUROCRYPT*, 2015, pp. 281–310.
[36] libsnark: a C++ library for zkSNARK proofs, [Online]. Available: https://github.com/scipr-lab/libsnark, Accessed: Oct. 2019.
[37] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von neumann architecture," in *Proc. USENIX Secur.*, 2014, pp. 781–796.
[38] Y. Zhang, C. Papamanthou, and J. Katz, "Alitheia: Towards practical verifiable graph processing," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2014, pp. 856–867.
[39] Y. Zhang, D. Genkin, J. Katz, D. Papadopoulos, and C. Papamanthou, "vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases," in *Prof. IEEE Symp. Secur. Privacy*, 2017, pp. 863–880.
[40] D. Liu, J. Ni, C. Huang, X. Lin, and X. Shen, "Secure and efficient distributed network provenance for IoT: A blockchain-based approach," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7564–7574, Aug. 2020.
[41] C++ library for Finite Fields and Elliptic Curves. [Online]. Available: https://github.com/scipr-lab/libff, Accessed: Oct. 2019.
[42] Parity Ethereum, [Online]. Available: https://github.com/paritytech/parity-ethereum, Accessed: Oct. 2019.
[43] Ethereum Status, [Online]. Available: https://etherscan.io, Accessed: Mar. 2020.
[44] F. Ye, Y. Qian, and R. Q. Hu, "A real-time information based demand-side management system in smart grid," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 2, pp. 329–339, Feb. 2015.

Dongxiao Liu (Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, in 2020. He is currently a Postdoctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include applied cryptography and privacy enhancing technologies for blockchain.



Jianbing Ni (Member, IEEE) received the B.E. and M.S. degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2014, respectively. He received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, Canada, in 2018. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON, Canada. His current research interests include applied cryptography and network security, with a focus on cloud computing, smart grid, mobile crowdsensing, and Internet of Things.



Xiaodong Lin (Fellow, IEEE) received the Ph.D. degree in information engineering from the Beijing University of Posts and Telecommunications, China, and the Ph.D. degree (with Outstanding Achievement in Graduate Studies Award) in electrical and computer Engineering from the University of Waterloo, Canada. He is currently an Associate Professor with the School of Computer Science at the University of Guelph, Canada. His research interests include computer and network security, privacy protection, applied cryptography, computer forensics, and software security.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. Dr. Shen is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Fellow, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.

Dr. Shen was the recipient of the R.A. Fessenden Award in 2019 from IEEE, Canada, Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) presents in 2019, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society, and Technical Recognition Award from Wireless Communications Technical Committee (2019) and AHSN Technical Committee (2013). He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He was the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'07, the Symposia Chair for the IEEE ICC'10, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. Dr. Shen is the elected IEEE Communications Society Vice President for Technical & Educational Activities, Vice President for Publications, Member-at-Large on the Board of Governors, Chair of the Distinguished Lecturer Selection Committee, Member of IEEE Fellow Selection Committee. He was/is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL, IEEE NETWORK, *IET Communications*, and *Peer-to-Peer Networking and Applications*.