Blockchain-based Secured IPFS-Enable Event Storage Technique with Authentication Protocol in VANET

Sanjeev Kumar Dwivedi, Ruhul Amin, and Satyanarayana Vollala

***Abstract*—In recent decades, intelligent transportation systems have improved drivers' safety and have shared information (such as traffic congestion and accidents) in a very efficient way. However, the privacy of vehicles and security of event information is a major concern, and blockchain technology has raised hopes that the problem of secure sharing of event information without compromising the trusted third party (TTP) and data storage issue can be resolved. Dwivedi et al. presented a blockchain-based protocol for secure sharing of events and authentication of vehicles. With the rigorous analysis of their protocol, it is found that only for secure storing of event information, they utilize the blockchain technology and authentication of vehicles solely depends on the cloud server. As a result, their scheme utilizes the notion of partially decentralized architecture. This article first shows the various loopholes of the blockchain-based event sharing protocol proposed by the Dwivedi et al. Then we propose a novel decentralized architecture for the vehicular ad-hoc network (VANET) without the cloud server, and based on it, the protocol for secure sharing of event information and vehicle's authentication using the blockchain mechanism has been proposed where the registered user access the event information securely from interplanetary file system (*IPFS*). We incorporate the *IPFS*, along with blockchain for storing the information in a fully distributed manner. Furthermore, the proposed protocol is compared with existing protocols, and the comparison provides desirable security at a reasonable cost. The evaluation of the proposed smart contract in terms of its associated cost is also presented in this paper.**

***Index Terms*—Blockchain, Secure information sharing, Authentication, *IPFS*, Security.**

I. Introduction

THE vision of Industry 4.0 is the digital transformation of industries based on innovative technologies such as cloud and cognitive computing, internet-of-things (IoT), blockchain, machine learning, etc. Industry 4.0 can provide more automation than the previous industrial revolution, and it bridges the gap between the physical and digital world. It helps to develop smart products and create a good ecosystem for them. Presently, various researchers and developers have put their efforts into developing the intelligent system. With the advent of IoT devices, more smart devices are developed, and these smart devices provide a foundation for the smart transportation system (STS). The security and safety of the drivers and efficiency of the transportation system are the main goals behind the STS. In this system, an intelligent vehicle is equipped with devices (such as sensors) which collect information and send it to roadside units (RSUs) and cloud servers. Traditionally, centralized architecture is used by the transportation system in which the required information is stored at a single place (such as a cloud server). The single-point-of-failure (SPoF) and the trust issue are the two major problems associated with this architecture. Moreover, the number of intelligent vehicles and users are increasing. As a result, the current system and the communication protocol cannot provide sufficient response to the system requirement [1].

Researchers are utilizing the distributed network architecture instead of a centralized architecture to provide trust and fairness in the system. Blockchain is a new distributed system, and Satoshi Nakamoto first implemented the underlying concept of blockchain in the famous bitcoin cryptocurrency [2]. Blockchain technology combines the ledger and consensus concept with peer-to-peer networking and security properties. Its architecture generally consists of six layers starting from the data layer (block structure), the network layer (block and transaction propagation), the consensus layer (block agreement), the incentive layer (reward), the smart contracts layer (business logic), and the application layer (programmable applications) [3]. In this regard, various researchers have tried to utilize the blockchain mechanism in various applications to solve problems in these systems.

Blockchain technology is combined with the existing vehicular ad-hoc network (VANET) protocols to solve the problems related to data sharing and vehicle authentication. In the existing system, data is shared among the intelligent vehicles and RSUs over non-secure channels. As a result, the probability data manipulation is high. In the continuation of it, Dwivedi *et al*. [4] presented a protocol for the secure sharing of event details among the RSU, and for vehicle authentication. Through close analysis of their protocol, we found that their protocol uses a centralized mechanism (cloud server) for vehicle authentication. The blockchain mechanism is used by them for secure storage of the event details. In this paper, we improve their protocol without involving a trusted

Manuscript received November 20, 2020; revised January 25, 2021; accepted March 14, 2021. This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

Recommended by Associate Editor .

Citation: S. K. Dwivedi, R. Amin, and S. Vollala, "Blockchain-based secured ipfs-enable event storage technique with authentication protocol in VANET," *IEEE/CAA J. Autom. Sinica*.

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

cloud server. The proposed solution uses the interplanetary file system (*IPFS*) and blockchain for storing the event details and for vehicle authentication.

A. Major Contributions

The main contributions of the paper include following

- In this paper, we present the weaknesses of Dwivedi's blockchain-based data sharing and authentication scheme and propose an improved protocol.
- To provide a secure and reliable sharing of event information in the vehicular network, the decentralized authentication protocol is designed.
- Along with the blockchain, the proposed mechanism also leverages *IPFS* for storing event information in a distributed manner.
- We provide the data accessing policies for the user based on the ethereum smart contract so that only the legitimate user can access the event details from *IPFS*.
- The consensus procedure for the common key is also presented in the paper.
- Our security analysis confirms that the proposed protocol is free from relevant attacks such as sybil attack, man-in-the-middle attack, etc., and achieves all the required security aspects.
- The various cost (computation, communication, and storage cost) incurred while proposing the protocol is computed, and compared with the Dwivedi et al. scheme.

B. Organization of the Papers

The rest of the paper is organized as follows: The related works for the blockchain-based data sharing in the vehicular network are presented in Section II. In Section III, we describe the loopholes of the Dwivedi et al. scheme, and provide solutions for overcoming the loopholes. The proposed architecture, decentralized authentication procedure, the user accessing policy, and consensus for the common key are discussed in Section IV. Section V presents the major security requirements and informal analysis of the proposed protocol. The performance analysis and comparisons with the existing scheme are discussed in Section VI followed by concluding remarks and future scope of the paper in Section VII.

II. Related Works

This section introduces existing research that utilizes the blockchain mechanism and its features (such as immutability, decentralization, etc.) to provide secure data sharing and to try to improve the privacy and security in communication protocols. Zhang *et al*. provide a data-sharing framework on the top of blockchain to tackle the announcing messages in internet-of-vehicles (IoV) [5]. Their validation is based on the proof-of-work, and to incentivize the vehicles, they further adopted the punish-reward mechanism. Authors in [6] discuss the message-dissemination problem and provide a blockchain-based solution to solve the problem. The delegated proof-of-stake consensus mechanism and reputation based algorithm for the selection of miners is used by the authors in [7]. The multi-weight subjective logic model is used by them for the computation of the vehicle's reputation. Yang *et al*. [8] presented a decentralized trust management scheme to achieve the vehicle's credibility. The proof-of-work and proof-of-stake approach with the bayesian inference mathematical model is used by them for the vehicle's message validation. The vehicle's identity and message reliability play a crucial role in the exchanging of messages with other vehicles and RSU in the IoV. To solve this, the authors proposed a consortium blockchain-based data sharing and storage system [9].

Further, the correctness and validation of traffic events is a tedious task in the current intelligent traffic management system. Yang *et al*. [10] presented the proof of event-based traffic event validation model. The events are collected by the RSU and submitted to the vehicles for validation purposes. To achieve trust, the authors proposed the authentication and entity-centric trust-building framework based on the blockchain [11]. The selection of the miner node is based on the trust value. The mobile edge computing and smart contract enabled blockchain is used by the authors for sharing the vehicular data [12]. The three-weight subjective logic model is utilized for the vehicle's reputation. Lun li *et al*. [13] presented the privacy-preserving incentive model for vehicles. The byzantine fault tolerance mechanism is used by authors for further validation of blocks, and both vehicles and RSU provide their consensus for it. Authors in [4] presented the secured event information sharing protocol based on the blockchain for the vehicular network. Their protocol uses the cloud server for vehicle authentication and blockchain for storing the event information among all the RSUs. The directed acyclic graph and permissioned blockchain-based asynchronous federated learning scheme are presented by the authors in [14] for the secure sharing of data in IoV. Moreover, a deep reinforcement learning approach is utilized by them for the selection of nodes and their optimization.

Authors in [15] utilize the proof-of-work consensus and vehicle's trust for solving the message dissemination problem in the VANET. Similarly, the reputation algorithm is used in [16] for solving the distribution of forged messages. Lu et. al. [17] utilizes the proof-of-absence and proof-of-presence based consensus protocol for solving the trust issues in VANET. On the other hand, the paillier cryptosystem is utilized by the authors in [18] for secure sharing of user's data. Huang et al. [19] proposed a blockchain and smart contracts (Stackelberg game) enabled framework to achieve the parked vehicle assisted fog computing services in a decentralized manner. Table I shows the research gap and blockchain specific challenge addressed by the existing solutions.

III. Loopholes in Dwivedi et. al.'s Scheme

This section describes the various drawbacks that are associated with the Dwivedi et al. scheme which are listed below:

1) Partially decentralized system: In the Dwivedi *et al*. scheme, roadside unit RSU submits the vehicle VEH registration details to the cloud server CS, and CS maintains the ledger, which stores the registration details for RSU and VEH. As a result, CS has all the details of the RSU and VEH which creates a partially decentralized network where all the

TABLE I
RESEARCH GAP AND BLOCKCHAIN SPECIFIC CHALLENGE ADDRESSED IN THE EXISTING BLOCKCHAIN-BASED SCHEME FOR VANET

S. No.	Reference	Blockchain specific challenge addressed	Research Gap
1	Ref [4]	Block validation protocol, smart contract	Partial centralized system, RSU storage capability constraint
2	Ref [5]	Smart contracts, mining incentive problem	Key generation and authentication protocol
3	Ref [6]	Proof-of-work consensus mechanism	Vechicle storage capability constraint
4	Ref [7]	Delegated Proof-of-stake consensus mechanism	Smart contract mechanism
5	Ref [8]	Proof-of-work consensus mechanism	Computation overhead
6	Ref [9]	Practical byzantine fault tolerance consensus mechanism, and smart contracts,	RSU storage capability constraint
7	Ref [10]	Proof-of-event consensus mechanism	Smart contract mechanism
8	Ref [11]	Miner selection for platoon blockchain	Consensus mechanism
9	Ref [12]	Consensus, smart contracts	Cloud storage capability constraint
10	Ref [13]	Byzantine fault tolerance (BFT) consensus mechanism	Improvement of BFT protocol
11	Ref [14]	Directed acyclic graph	RSU storage capability constraint
12	Ref [15]	Proof-of-work consensus mechanism	Smart contract mechanism.
13	Ref [16]	Proof-of-work consensus mechanism	Smart contracts mechanism
14	Ref [17]	Proof-of-absence, and proof-of-presence protocols	Smart contracts mechanism
15	Ref [18]	Smart contracts	Cloud storage capability constraint

sensitive information of the system is stored in a single place.

2) Single-point-of-failure (SPoF) problem: The entire functioning of Dwivedi et al. scheme is based on the availability of the information provided by the CS. If somehow, the CS was offline (or crashed), their system will not work. Hence, we conclude that the SPoF problem is still present in their system.

3) Centralized authentication mechanism: In the Dwivedi *et. al*. scheme, CS performs authentication of VEH. To perform this, VEH sends the event information E_m with the necessary parameters (such as $\langle ID_{VEH}, ID_{RSU}, E_m \rangle$) to RSU. RSU simply forwards these details to the CS. Now, the CS does the computation and provide its feedback to the respective RSU; whether the VEH authentication is successful or not. Hence, we conclude that a centralized authentication mechanism is used by them.

4) Storage capability constraint: In the Dwivedi *et al*. scheme, they assume that the $RSUs$ have enough storage space. Therefore it can store the event-information (both data and hash) of any length which is further based on the blockchain mechanism.

A. Solutions for Overcoming the Loopholes of Dwivedi et. al.'s Scheme

This section briefly discusses how the proposed methodology overcomes the loopholes of the Dwivedi et al. scheme. Each point is discussed below:

1) Fully decentralized system: The blockchain-based solution provided by Dwivedi et al. utilizes a third party as a cloud server CS which stores the crucial information of VEH and RSU. We further enhance their scheme and proposed a new system that is completely free from the CS. In our proposed system, both the VEH and event information are stored in a decentralized manner. Therefore, a fully decentralized system is provided by our scheme.

2) No SPoF problem: Furthermore, the Dwivedi et al. scheme suffers from the SPoF problem because crucial information of VEH is available in a single place. Our proposed scheme utilizes blockchain for storing the VEH information in a completely decentralized manner. Therefore, our scheme never suffers from a SPoF problem.

3) Decentralized authentication mechanism: In the Dwivedi et al. scheme, the CS performs the authentication of VEH. But, we suggest a novel methodology in which the VEH information is distributed among all the available RSU of the system. Therefore in our scheme, any RSU can perform authentication of VEH using the blockchain.

4) No storage capability constraint: To resolve the storage capability constraint of the Dwivedi et al. scheme, our proposed protocol uses the $IPFS$ with the blockchain mechanism that can resolve the RSU storage problem. $IPFS$ stores the information across the globe. For this, we further provide the user accessing rules based on the smart contract.

IV. PROPOSED SYSTEM

A. System Overview

In this section, we try to improve Dwivedi et al.'s protocol without including the cloud server as a trusted third party (TTP). As shown in Fig 1. the proposed blockchain-enabled protocol consists of network administrator NA, roadside unit RSU, vehicle VEH, and blockchain network with the $IPFS$. All the notations that are used throughout the paper are illustrated in Table II. A brief description of the entities are given below:

Network Administrator $\langle NA \rangle$: In the proposed protocol, NA performs the registration of RSU. Before the deployment of $\langle RSU \rangle$ in the network, NA does the off-line registration and stores the common key $\langle K \rangle$ in all the $RSUs$. This key $\langle K \rangle$ is used by the RSU for exchanging the *hash* that it gets from IPFS. The key $\langle K \rangle$ is valid only for the fixed duration of T.

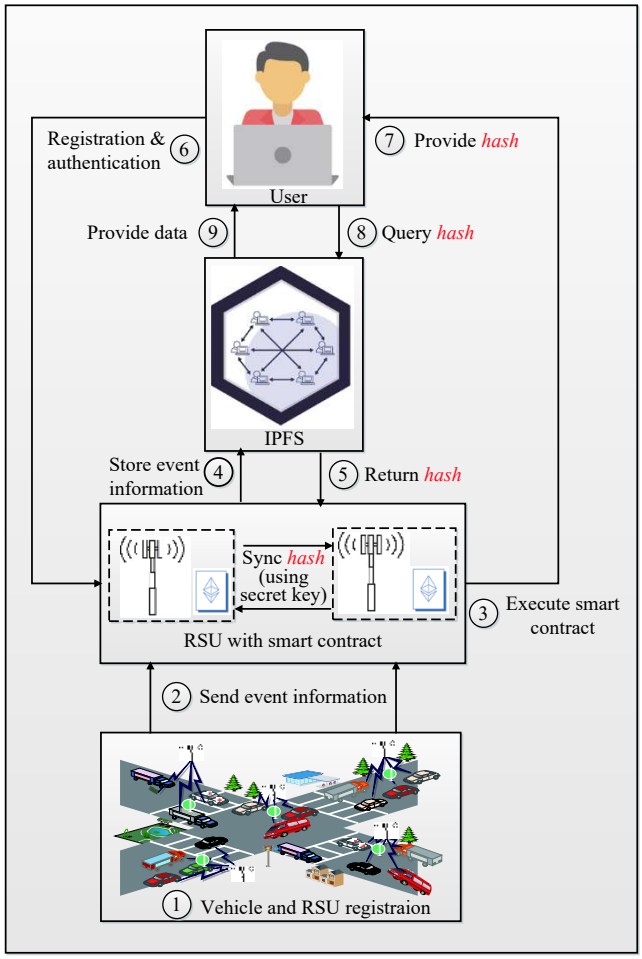

Fig. 1. Proposed Architecture[1].

After the time T, any RSU must execute the generation and consensus algorithm for the common key and share the new key K_1 to all $RSUs$.

Roadside Unit $\langle RSU \rangle$: $RSUs$ perform the registration of vehicle VEH. For every three or four kilometres, one RSU is present, and one VEH needs to register with only one RSU. After the successful registration of VEH, RSU selects a few registration parameters, creates a block $\langle B_1 \rangle$, and executes the block-validation procedure. Once block validation is successful, the RSU hands over the index of block $\langle B_1 \rangle$ (i.e., I_{B_1}) to VEH. Now for further authentication of VEH, it needs to send I_{B_1} along with the registration parameters.

Vehicle $\langle VEH \rangle$: The VEH is equipped with the on-board unit (OBU), Wi-Fi, a global positioning system (GPS), etc. The computational and storage capability of VEH is very low, and it stores only a few sensitive parameters such as an identity of RSU ID_{RSU}, the identity of VEH TID_{VEH}, A, B, etc. The VEH collects events E_m and forwards it to the nearest RSU for further processing.

Blockchain and $\langle IPFS \rangle$: The proposed work mainly focuses on the decentralized authentication and distributed storage of data (events) based on *IPFS* and blockchain. *IPFS* uses the content addressable methodology that can solve the blockchain storage problem. The main idea of *IPFS* is to store the encrypted data or file in decentralized nodes and on the behalf of this, *IPFS* provides the hash of encrypted data. This hash is used to retrieve the encrypted data from *IPFS*.

Smart Contracts: Unlike traditional contracts, smart contracts are computerized programs which are executed automatically once the condition of the contracts among the peers (nodes) is satisfied. It runs on a decentralized platform (such as an ethereum-based blockchain) and as a result, a third party is not required for executing the transactions (or assets), and its verification. The smart contracts are immutable, and they are cryptographically secure. Our proposed methodology also utilizes the ethereum enabled smart contracts for securely storing the event details in the *IPFS*, and further accessing the event details from *IPFS* through the RSU which is shown on Algorithm 1. By doing this, the immutable event details are present in the *IPFS* so that in the future, if any authentic user wants to access event details form blockchain enable *IPFS* servers, he/she can access it once the smart contract conditions are satisfied.

TABLE II
DESCRIPTIONS OF VARIOUS NOTATIONS USED IN PROPOSED SCHEME

Notations	Description
NA	Network Administrator
RSU_a	Roadside Unit (a)
VEH_b	Vehicle (b)
USR_i	User (i)
ID_{RSU_a}	Identity of RSU_a
ID_{VEH_b}	Identity of VEH_b
TID_{VEH_b}	Temporary Identity of VEH_b
PU_{RSU_a}	Public key of RSU_a
PR_{RSU_a}	Private key of RSU_a
PKG	Public key Generator
N_{VEH_b}	Nonce Generated by RSU_a for VEH_b
K & $K1$	Common key among the RSU for exchanging the *hash*
hash	Hash provided by the *IPFS* to RSU_a
C_{VEH_b}	Challenge for VEH_b
R_{VEH_B}	Response for VEH_b
PUF	Physical Unclonable Function
E_m	Event (m)
B_i	New Block (i)
I_{B_i}	Index for Block (i)
$h(\cdot)$	Cryptographic Hash Function
$E(\cdot)$	Encryption Function
$D(\cdot)$	Decryption Function
$\|\|$	Concatenation Operator

[1] No.1 indicates the vehicles and RSU registration performed by the RSU and network administrator respectively. No.2 indicates the one of the vehicle collects the event information using its sensing units, and send it to the nearest RSU for further processing. No.3 indicates RSU validates the event information and execute the smart contracts for creating the new block. No.4 and 5 indicate that verified event information is published on IPFS, and in return, IPFS provides the hash of that event to RSU. No.6 indicates the user registration and authentication procedure. No.7 indicates the after the successful authentication of the user, RSU provides the hash of the event to the user. No.8 and 9 indicate the user request for event details from IPFS and in return IPFS provides the event detail to the user.

Our proposed scheme is divided into several phases; started from the registration phase where vehicle and RSU registration is done by the RSU and NA, respectively. The event collection E_m and $VEHs$ authentication is the second phase where authentication of VEH is performed by the near-by RSU based on the blockchain. The detailed description of all the phases are given in the Subsection 4.2, 4.3, 4.4, 4.5.

B. Registration Phase

a) Roadside unit (RSU) Registration

The following steps are executed by the RSU_a.

Step 1: Roadside unit RSU_a chooses a unique identity $\langle ID_{RSU_a} \rangle$, and sends it to NA using the secure channel.

Step 2: After receiving $\langle ID_{RSU_a} \rangle$, NA generates the public and private key $\langle PU_{RSU_a}, PR_{RSU_a} \rangle$ for the RSU_a. The public key generator (PKG) is used by the NA for the generation of key-pair.

Step 3: The NA delivers $\langle PUBRSU_a, PR_{RSU_a} \rangle$ to RSU_a through off-line (or) by using the secure channel.

Step 4: Once RSU_a gets $\langle PUBRSU_a, PR_{RSU_a} \rangle$, it publicly declares $\langle PU_{RSU_a} \rangle$ so that other RSU of the system also knows $\langle PU_{RSU_a} \rangle$ of RSU_a, and it securely keeps $\langle PR_{RSU_a} \rangle$.

Step 5: Before the deployment of RSU in the system, it should agree on one common key K. The procedure for the generation and consensus on key K is discussed in Subsection 4.7.

b) Vehicle (VEH) Registration

The following steps are executed by the VEH.

Step 1: User of the vehicle VEH_b chooses a unique identity $\langle ID_{VEH_b} \rangle$, sends the $\langle ID_{VEH_b} \rangle$ and bio-hashing of the biometric information of the user $\langle H(B_{info}) \rangle$ to the nearest RSU (consider RSU_a as the nearest RSU) through off-line or in-person.

Step 2: On receiving $\langle ID_{VEH_b} \rangle$ and $\langle H(B_{info}) \rangle$, RSU_a computes temporary identity $\langle TID_{AV_b} \rangle$ as in equation 1.

$$TID_{VEH_b} = h(ID_{VEH_b} \| N_{VEH_b}) \tag{1}$$

Step 3: After that, it computes the parameter A_{VEH_b} using the $\langle TID_{VEH_b}, H(B_{info}) \rangle$ as in equation 2.

$$A_{VEH_b} = h(TID_{VEH_b} \| H(B_{info})) \tag{2}$$

Step 4: Now, RSU_a selects challenge C_{VEH_b}, and it uses PUF to get R_{VEH_b} corresponding to C_{VEH_b}.

Step 5: RSU_a stores $\langle TID_{VEH_b}, ID_{RSU_a}, C_{VEH_b}, PUF, A_{VEH_b} \rangle$, in the OBU of VEH_b.

Step 6: Once step 5 is over, RSU_a creates a new block B_1 by using the parameters $\langle A_{VEH_b}, ID_{RSU_a}, C_{VEH_b}, R_{VEH_b} \rangle$. The block creation phase is discussed in Subsection 4.4.

C. Transaction-generation (Event Detection) Phase

In this phase, one of the vehicle VEH collects the critical events (e.g., accident on road, traffic congestion, etc.) through its OBU and Wi-Fi units, and sends it (along with TID_{VEH_b}, ID_{RSU_a}) to the nearest RSU. That RSU checks the integrity of the event and confirms the necessary action accordingly. We consider that the event E_m is collected by the VEH_b and it forwards E_m to the RSU_a.

Step 1: VEH_b computes $B = h(E_m \| R_{VEH_b})$ (using OBU), and sends the parameters $\langle TID_{VEH_b}, ID_{RSU_a}, A_{VEH_b}, C_{VEH_b}, R_{VEH_b}, B, E_m, I_{B_1} \rangle$ to RSU_a which is encrypted $E(.)$ using the PU_{RSU_a}. PUF is used by VEH_b for acquiring the R_{VEH_b}.

Step 2: Once RSU_a gets the encrypted parameters, RSU_a decrypts $D(.)$ using the PR_{RSU_a} to get the original parameters.

Step 3: Now, RSU_a computes the $B' = h(E_m \| R_{VEH_b})$, and checks whether $(B' == B)$?

Step 4: If the condition $(B' == B)$ is correct, RSU_a predicts that the event E_m provided by the VEH_b is correct. No adversary changes E_m in the public channel.

Step 5: After the correctness of E_m, RSU_a checks the authenticity of vehicle VEH_b by looking-up the blockchain which is discussed in Subsection 4.5.

D. Block Creation Phase

In this phase, RSU_a creates blocks $\langle B_1, B_2 \rangle$, which is based on the proof-of-work consensus mechanism[2]. The parameters $\langle A_{VEH_b}, ID_{RSU_a}, C_{VEH_b}, R_{VEH_b} \rangle$ are used for the computation of merkle root $root_{mer}$ of B_1, and for B_2, parameters $\langle TID_{VEH_b}, E_m \rangle$ are used. The $root_{mer}$ is stored in the block header for the computation of current block hash $hash_{cur}$ as in equation 3.

$$hash_{cur} = h(hash_{pre} \| root_{mer} \| N \| TS_i) \tag{3}$$

where for block B_1, $(root_{mer})$ is computed as:

$(root_{mer}) = h(H_1 \| H_2)$.

$H_1 = h(h(A_{VEH_b}) \| h(ID_{RSU_a}))$[3].

$H_2 = h(h(C_{VEH_b}) \| h(R_{VEH_b}))$[4].

and for block B_2, $(root_{mer})$ is computed as:

$(root_{mer}) = h(H_3 \| H_4)$.

$H_3 = h(TID_{VEH_b})$[5].

$H_4 = h(E_m)$[6].

$hash_{pre}$ = Hash of previous block

N = Nonce used for computing the hash value of current block

TS_i = Time-stamp

Initially, N is set to zero, and at each iteration, it is incremented by one. The block values $(hash_{pre})$, $(root_{mer})$, and (TS_i) are repeatedly hashed with different values of N for the computation of current hash index $(hash_{cur})$. After the block creation is successful, RSU_a hands over the index of block B_1 (i.e., I_{B_1}) to the respective vehicle VEH_b. As a result, I_{B_1} is used by the VEH to further the authentication

[2] Proof-of-work is the mechanism that provides the consensus (common agreement) among the nodes of the decentralized network. Its principle is based on a solution (current hash of a block) that is difficult to find, but at the same time verification is easy for that solution. Here, the nodes of the decentralized network are continuously engaged in finding a nonce value which, when hashed with the previous block hash with necessary parameters produces a resultant lesser than the predefined threshold.

[3] H_1 is a hash value, which is computed by hashing pairs of A_{VEH_b} and ID_{RSU_a}.

[4] H_2 is a hash value, which is computed by hashing pairs of C_{VEH_b} and R_{RSU_a}.

[5] H_3 is a hash value, which is computed by hashing of TID_{VEH_b}.

[6] H_4 is a hash value, which is computed by hashing of E_m, whereas $h(.)$ is one way hash function (eg. SHA-256).

procedure. The block B_2 is securely stored in the blockchain that is maintained by RSU.

E. Vehicle Authentication and Event Storage Procedure

In Subsection 4.3, we consider that the VEH_b collects E_m and sends parameters $\langle TID_{VEH_b}, ID_{RSU_a}, A_{VEH_b}, C_{VEH_b}, R_{VEH_b}, B, E_m, I_{B_1}\rangle$ to RSU_a. The procedure for vehicle authentication is illustrated below:

Step 1: Once RSU_a receives parameters $\langle TID_{VEH_b}, ID_{RSU_a}, A_{VEH_b}, R_{VEH_b}, B, E_m, I_{B_1}\rangle$, it uses I_{B_1} and searches the $root_{mer}$ corresponding to the I_{B_1}. Constant time (i.e., $o(constant)$) is required for the searching of $root_{mer}$.

Step 2: After getting the $root_{mer}$, RSU_a recomputes the $root'_{mer}$ based on the parameters it receives. The equation 4. illustrates the computation of $root'_{mer}$.

$$(root'_{mer}) = h(H_5 \,\|\, H_6) \tag{4}$$

where $H_5 = h(h(A_{VEH_b}) \,\|\, h(ID_{RSU_a}))$.

$H_6 = h(h(C_{VEH_b}) \,\|\, h(R_{VEH_b}))$.

Step 3: After the computation of $root'_{mer}$, RSU_a checks whether $(root'_{mer} == root_{mer})$?

Step 4: If condition $(root'_{mer} == root_{mer})$ holds, then RSU_a believes that VEH_b is an authentic vehicle.

Step 5: After the successful authentication of VEH_b and correctness of E_m(as described in Subsection 4.3), RSU_a stores the event in the decentralized nodes of *IPFS*.

Step 6: On behalf of the uploaded event E_m on *IPFS*, it provides *hash* of E_m to RSU_a.

Step 7: RSU_a stores this *hash* in its ledger, and distributes *hash* to other $RSUs$ (using the common key K).

Step 8: RSU_a selects nonce N_a, and computes the $PR1_{RSU_a}$ as in equation 5

$$PR1_{RSU_a} = h(PR_{RSU_a} \,\|\, N_a) \tag{5}$$

Step 9: RSU_a encrypt $E(.)$ the *hash*, which is exchanged among RSU (in step 7) by using $PR1_{RSU_a}$, and stores the encrypted hash in its local ledger.

Step 10: The same process (step 8 & 9) is repeated by all the $RSUs$ for storing the hash in their ledger. Also, the $PR1_{RSU}$ needs to be stored securely so that no one can know about the $PR1_{RSU}$.

F. Accessing Stored Event E_m from IPFS

This section presents the smart contract algorithm which is developed and implemented as a solution for accessing the stored event E_m from *IPFS* through the RSU.

Algorithm 1: Smart contract algorithm for accessing E_m from *IPFS* through RSU_a

Input: ID_{RSU_a}, $EA(USR_i)$, $EA(RSU_a)$
Output: *hash*, E_m
1: Parameters:
USR_i: User(i)
RSU_a: Road-side unit(a)
ID_{RSU_a}: Identity of RSU(a)
$EA(USR_i)$: Ethereum address of user(i)
$EA(RSU_a)$: Ethereum address of RSU(a)
TKN: Token $(EA(USR_i), ID_{RSU_a})$
BC: Blockchain
E_m: Event(m)
hash: Hash of E_m
2: begin
3: **if** ($EA(RSU_a)$ exists in (BC) = True) **then**
4: **if** $EA(USR_i)$ exists in (BC) = False) **then**
5: create $TKN(EA(USR_i), ID_{RSU_a})$;
6: **else**
7: display: USR_i is already registered with RSU_a;
8: **end iffont-size:9pt**
9: return error;
10: **end if**
11: **if** ($EA(RSU_a)$ exists in (BC) = True) **then**
12: **if** $EA(USR_i)$ exists in (BC) = True) **then**
13: **if** (validate $(TKN(EA(USR_i), ID_{RSU_a}), BC)$ = True) **then**
14: display: USR_i is authenticated successfully;
15: **else**
16: display: authentication of USR_i is not successful;
17: **end if**
18: display: USR_i is not registered;
19: **end if**
20: return error;
21: **end if**
$USR_i \leftarrow hash$;
$USR_i \leftarrow$ search (*hash*, *IPFS*);
$USR_i \leftarrow E_m$;
22: end begin;

The smart contract is written in solidity language and tested in Remix IDE. The RSU and User USR_i are the two main actors of our proposed contract, and each is identified via an ethereum address (EA). The users are registered in the system through the smart contract, and every USR_i is linked with a unique ethereum address (EA). After the successful registration of the USR_i, a unique token TKN is created. The TKN is the combination of a user ethereum address ($EA(USR)$) and identity of RSU ($ID(RSU)$). This TKN is used in the later stages for interacting with the contract.

If the end-user USR_i (e.g., police: verifying suspicious activity) wants to access the data (or event E_m), the USR_i sends the essential parameters (such as the identity of a vehicle, timestamp of event, ethereum address of USR_i, and RSU) required for the authentication of the smart contract. To authenticate the USR_i, the following conditions must be satisfied: (1) $EA(RSU)$ exists in the contract, (2) $EA(USR)$ exists in the contract, and (3) there exists a valid TKN in the contract. If the above-mentioned conditions are satisfied, then the contract confirms that the USR_i are authenticated successfully. After the successful registration and authentication of USR_i, RSU provides the *hash* of the desired event E_m to the USR_i. The USR_i invoke a query (it seeks to obtain the original event E_m that corresponds to this *hash*) from *IPFS*, and in return, *IPFS* provides the desired event E_m to the concerned user. The procedure for accessing policy of E_m through *IPFS* for USR_i is illustrated in Algorithm 1, and smart contract evaluation for the same algorithm is discussed in Subsection 6.1.

G. Consensus Procedure for Common Key K

As discussed in Subsection 4.2, the $RSUs$ should agree on the key K. This agreement is desirable for our system because *IPFS* provides *hash* to initiator RSU on behalf of the stored event, and all $RSUs$ must exchange it using K. To provide secure communication between $RSUs$ and to provide better security for the system, the authentication mechanism for the RSU is also illustrated in this Section.

Step 1: At the time of deployment to the RSU in the system, *NA* stores randomly the N number of hashes in one RSU. We consider that *NA* store it on RSU_a.

Step 2: RSU_a creates a merkle tree from the stored N number of hashes and securely stores the root value of merkle tree $root_{mer}$. The RSU_a also creates a genesis block (first block of blockchain) from the computed $root_{mer}$ as in equation 3, where $hash_{pre}$ is all zero values for the genesis block.

Step 3: Now, RSU_a randomly selects M number of hashes from the stored N number of hashes (where $M \leq N$). Once selection procedure is over, RSU_a concatenate M number of hashes and sends it to the other RSU which is encrypted $E(.)$ using the public key of RSU (PU_{RSU}).

Step 4: Other $RSUs$ of the system decrypt $D(.)$ it using their private key PR_{RSU}, and compute the common key K including the RSU_a as in equation 6.

$$K = h[H_1 \,||\, H_5 \,||\, H_7 \,||\, ... \,||\, H_M \,||\, T] \tag{6}$$

where H_1 and H_M are selected hashes from N number of hashes, and T is the current time-stamp.

Step 5: After the computation of K, the initiator RSU_a computes the new parameters S_1 and S_2 and sends S_2 to the other $RSUs$ of the system.

$S_1 \leftarrow E_{PR_{RSU_a}}(ID_{RSU_a} \,||\, K)$.

$S_2 \leftarrow E_K(S_1)$.

where PR_{RSU_a} is the private key of RSU_a (provides digital-signature), and K is the computed common key.

Step 6: Once other $RSUs$ receive the parameter S_2, they decrypt $D(.)$ it by using the key K since they already computed K in step 4.

$S_1 \leftarrow D_K\,[E_K(S_1)]$.

Step 7: After that RSU decrypts $D(.)$ S_1 by using public key of PU_{RSU_a}, and checks whether the received K is the same as the computed K or not.

$(ID_{RSU_a} \,||\, K) \leftarrow D_{PU_{RSU_a}}\,[E_{PR_{RSU_a}}(ID_{RSU_a} \,||\, K)]$.

Step 8: Steps 1 through 7 ensure that all $RSUs$ agree on the common key K, and this key is used for the exchanging of *hash* which is provided by *IPFS*. Moreover, the identity of RSU_a (i.e., ID_{RSU_a}) is also present in the encrypted message S_1. Therefore, it confirms that RSU_a is an authentic RSU.

a) Procedure for Generating the New Key K_1

As discussed in Subsection 4.7, $RSUs$ agree on the key K for exchanging the *hash*. But, the key must be updated on a regular interval. The key was exchanged among the $RSUs$ using public and private keys $\langle PU_{RSU},\ PR_{RSU}\rangle$ which are generated by the *NA*. Therefore, there may be a chance that *NA* can disclose the PR_{RSU} to a third party, and as a result, the third party can maintain its ledger for storing the *hash* provided by the *IPFS*. To solve this issue, K must be changed after a certain period. Here, we assume that the RSU_a wants to create a new common key K_1. To do this, RSU_a adds the previous key K into the pool of existing N number of hashes and randomly selects M number of hashes (where $M \leq N+1$). After that, for creating the new common key, the same procedure is repeated as discussed in Subsection 4.7 (from step 3 to step 8). The only difference is that for the computation of parameters S_1 and S_2, this time key K_1 is used instead of key K, and $E(.)$ and $D(.)$ operations are performed by using the K_1. Once consensus is achieved for K_1, it is used by the RSU for exchanging the *hash*, provided by the *IPFS*.

V. Security Requirements and Informal Analysis

In order to meet the effective operations of VEH and RSU, the necessary security requirement such as authentication, integrity, and availability must be achieved by the system. In this section, we first discuss how the proposed scheme achieves these security requirements, and then discuss the relevant security attacks.

Authentication, Integrity & Availability: The parameters for creating a block for authentication of VEH_b $\langle A_{VEH_b}, ID_{RSU_a}, C_{VEH_b}, R_{VEH_b}\rangle$ (i.e., $root_{mer}$) are permanently stored in the blockchain, as discussed in Subsection 4.4. Therefore, any RSU of the system performs the authentication of VEH_b. Moreover, VEH_b computes $B = H(E_m \,||\, R_{VEH_b})$ and sends parameters $\langle TID_{VEH_b}, ID_{RSU_a}, A_{VEH_b}, C_{VEH_b}, R_{VEH_b}, B, E_m, I_{B_1}\rangle$ to RSU_a. In order to check the integrity of E_m, RSU_a recomputes $B' = H(E_m \,||\, R_{VEH_b})$ and verifies the condition $(B' == B)$. In addition to this, events are stored in the blockchain, and it is always available in *IPFS*.

Sybil Attack: Under the sybil attack, an adversary creates multiple fraudulent identities and controls them in a decentralized peer to peer network. The objective is to segregate the target node from the rest of the honest nodes so that it can gain a major influence in the network, carry out illegal activities, and launch different attacks such as double-spending, refuse the transactions and blocks, etc. In the proposed protocol, VEH_b has a unique identity $\langle IID_{VEH_b}, TID_{VEH_b}\rangle$, and its identities are not shared with other $VEHs$ of the system. Before the sharing of event details with RSU_a, VEH_b is first authenticated using the blockchain (I_{B_1}). The VEH_b is not allowed to acquire multiple identities, and anyhow, if VEH_b gets the more than one identity, the required condition for authentication ($root'_{mer} == root_{mer}$) is not satisfied. Therefore, launching the sybil attack from an adversary perspective is almost impossible.

Message Substitution Attack: In the proposed protocol, VEH_b sends the event details to RSU_a, which mainly uses the hashing and encryption operations. The event detail E_m is hashed with the parameter R_{VEH_b}, encrypted using PU_{RSU_a}, and then sends the encrypted parameter to RSU_a. Therefore, if an adversary somehow gets R_{VEH_b}, the adversary can't get the original transaction due to the pre-image resistance property of hash functions, and the required condition $(B' == B)$ is also checked during the transaction (or message) generation and authentication phase. As a result, a message substitution attack is not possible in the proposed protocol.

Man-in-the-Middle Attack: Assume that an adversary intercepts an authentication message which is sent by VEH_b

TABLE III
COMPUTATION, COMMUNICATION AND STORAGE COST OF PROPOSED PROTOCOL AND COMPARED WITH REF.[4][7]

S. No.	Computation Cost		Communication Cost (bits)		Storage Cost (bits)	
	Proposed	Ref. [4]	Proposed	Ref. [4]	Proposed	Ref. [4]
P-I	$1T_{pf} + 1T_{pg} + 3T_h$	$2T_{pf} + 2T_{pg} + 1T_e + 1T_d$	768	896	1408	2048
P-II	$10T_h + 1T_e + 1T_d\ 1T_{\delta_1} + 1T_{\delta_2}$	$4T_h + 4T_e + 1T_c + 2T_{pf}\ 1T_{\delta_1} + 4T_{\delta_2}$	2304	2816	256	256

to RSU_a in the message generation and authentication phase, and that it uses a third party to perform this attack. Since the proposed protocol uses the blockchain for authentication of VEH_b, and the sensitive identity information $\langle A_{VEH_b}, ID_{RSU_a}\rangle$ is permanently stored in the blockchain, and it is distributed over several $RSUs$ which are linked to the previous block. Therefore, if the third party publishes the fake keys, it has to change the entire chain which is computationally not feasible, and easily traceable by other $RSUs$. Thus, this scheme can resist the man-in-the-middle attacks.

Blockify Attack: Under the blockify attack, an adversary intentionally steals the others block information from the public (or) private blockchain and wants to add this information into its blockchain. The objective of an adversary is to create the same replica of blockchain so that it can carry out illegal activities. Assume that the third party TP somehow gets the new block B_i and it wants to add this block into its blockchain network. Since TP knows only the partial information from the public channel, it cannot recompute the $hash_{cur}$. TP does not know the value of $hash_{pre}$. Moreover, if TP willfully wants to know $hash_{pre}$, it has to change the entire chain of blocks, which is computationally infeasible. Hence, the proposed scheme secure against the blockify attack.

VI. PERFORMANCE ANALYSIS AND COMPARISON

In this section, we present the performance of our scheme and compare it with the Dwivedi et al. scheme. The security properties (such as computation and storage cost of the node, communication cost between two nodes) are used for measuring the performance of both schemes. For measuring and comparing this cost, we considered only one vehicle VEH_b and one roadside unit RSU_a. The proposed approach mainly uses the encryption $E(.)$ and decryption $D(.)$ operations along with the lightweight hash function $h(.)$. $256-bit$ hash function is used for the computation of T_h. PKG is used (the first time only) for the generation of keys of RSU. For the comparison with the existing scheme, we consider that the length of vehicle and RSU identity (i.e., ID_{RSU_a}, ID_{VEH_b}), and parameters $\langle TID_{VEH_b}, A_{VEH_b}, C_{VEH_b}\rangle$ each takes 128 bits. Table III provides the computation, communication, and storage costs of our proposed scheme, and the same is compared with the Dwivedi et al. scheme.

The proposed scheme utilizes the key K for the exchange of *hash* (provided by the *IPFS*) among the RSU. The various costs such as computation, communication, and storage cost involve in the consensus procedure of key is $[(2N-1)T_h + (2M-1)T_h + 2T_e + 2T_d]$, $[(M*256) + 128]$ bits, and $[(N*256) + 128]$ bits respectively, where N indicates the total number of hashes stored in RSU_a and M indicates the number of hashes picks in the current iteration for generating the K, and $M \le N$. Fig. 2 presents the storage cost (in bytes) of the proposed protocol and protocols in [4], [20], [21]. It is noted that our proposed protocol achieves better performance than the existing protocols.

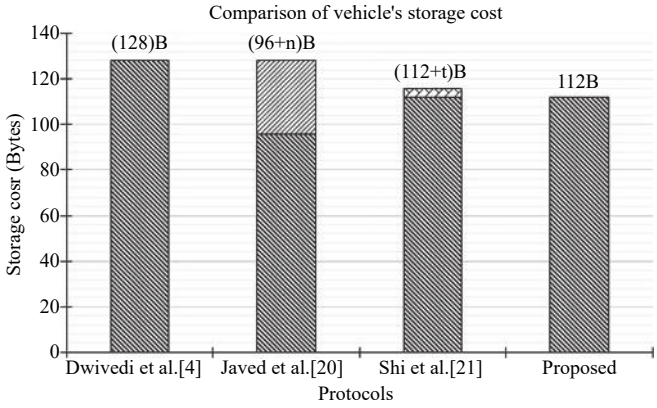

Fig. 2. Comparison of Vehicle's Storage Cost with Existing Scheme[8].

A. Remix IDE and Solidity-based Evaluation of Smart Contract

The smart contract proposed in Algorithm 1 for accessing the E_m from IPFS through RSU is evaluated by writing it in the solidity language. The identity of RSU_a, ethereum address of USR_i, and ethereum address of RSU_a are inputs, whereas $event_m$ and hash of that event are output for the Algorithm 1. Symbolically inputs and outputs are represented by $\langle ID_{RSU_a}, EA(USR_i), EA(RSU_a)\rangle$ and $\langle E_m, hash\rangle$, respectively. The REMIX IDE and Solidity version 0.5.10 with a specification of Win10, 8GB of RAM, Intel(R) Core(TM) i5-8250U CPU @1:60GHz, 64- bit OS is used.

a) Deploying Cost:

The proposed smart contract presented in the Algorithm 1 is deployed in the JavaScript VM environment. *0xddaad340b0-f1ef65169ae5e41a8b10776a75482d* is the smart contract address, and *0x5B38Da6a701c568545dCfcB03FcB875f56-beddC4* is the account address. The same account address is used for deploying the smart contract under different intervals. In solidity, every execution step has a cost (transaction and execution cost) associated in terms of GAS. Fig. 3 shows the analogy of the cost of deploying the proposed smart contracts

[7] P-I: Registration of vehicle and RSU; P-II: Event generation and authentication; T_{pg}: Time required by PKG function; T_{pf}: Time required by PUF; T_e: Time required to encrypt parameters; T_d: Time required to decrypt parameters; T_h: Time required by One-way hash function; T_{δ_1}: Time required for searching the parameters in ledger; T_{δ_2}: Time required for matching the required conditions;

[8] In Javed et al. and Shi et al. approach, n and t is the cryptoid of vehicles, and time-stamp respectively. For comparison, we consider n and t is length of 32 bytes and 4 bytes.

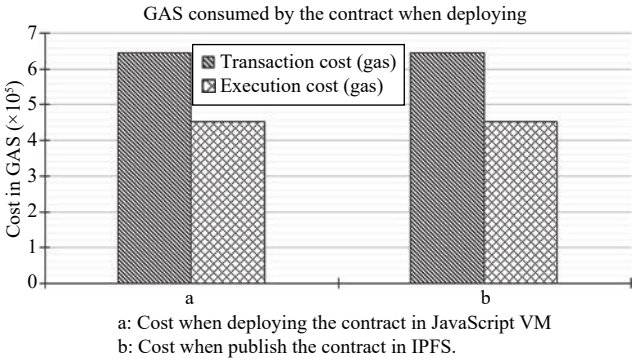

Fig. 3. Deploying Cost of Smart Contract.

in JavaScript VM and publish the same to the *IPFS*. Interestingly, the same cost is consumed by the contract.

b) Transaction Cost vs Execution Cost:

Users first do the registration in our system through smart contracts if they want to access the stored data from *IPFS*. The smart contract should be designed in such a way that we can analyze the transaction and execution cost to understand how our system behaves with an increasing amount of data stored in the *IPFS*. User registration and its authentication, and fetching the stored data from *IPFS* all are variable and independent process. Variable means that the size of event information varies and independent means that fetching the stored data from *IPFS* are not dependent on each other. Fig. 4 shows the analogy of execution cost and transaction cost (in terms of GAS) of proposed smart contracts. The cost required for sending the program to a blockchain is known as transaction cost, whereas the cost required to execute the computational operations is known as execution cost. For the accurate analysis of transaction and execution cost, the variable number of users are registered in the system using the smart contracts. But, interestingly, each time the rate of growth of the curve is the same, and the transaction costs are much higher than the execution costs.

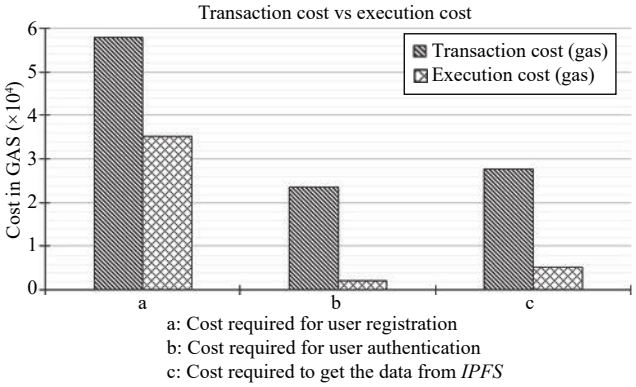

Fig. 4. Transaction Cost and Execution Cost of Smart Contract.

VII. Concluding Remarks & Future Scope

This article presents the blockchain-based decentralized architecture for secure sharing of event information among roadside units without considering trusted third parties (cloud server). In this paper, we examined the Dwivedi et al. blockchain-based protocol and illustrated the weaknesses of their protocol. To address the Dwivedi et al. security weaknesses, we proposed the improved event sharing and vehicle authentication protocol based on the *IPFS* and blockchain. Our protocol utilizes the *IPFS* for distributedly storing the information, and authenticate the vehicle using the blockchain. The consensus mechanism for the common key is also presented in the paper. The security analysis of the protocol confirms that protocol is free from possible attacks, and achieves security requirements. We also computed the computation, communication, and storage cost of the proposed protocol, and compared it with the Dwivedi et al. protocol. The performance section confirms that the vehicle's storage cost is less than the existing one. In future research we will work more on the smart contracts mechanism and continue enhance the security of our proposed scheme.

References

[1] Zhaojun Lu, Gang Qu, and Zhenglin Liu, "A survey on recent advances in vehicular network security, trust, and privacy," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 760–776, 2018.

[2] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[3] Peiyun Zhang and Mengchu Zhou, "Security and trust in blockchains: Architecture, key technologies, and open issues," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 3, pp. 790–801, 2020.

[4] Sanjeev Kumar Dwivedi, Ruhul Amin, Satyanarayana Vollala, and Rashmi Chaudhry, "Blockchain-based secured event-information sharing protocol in internet of vehicles for smart cities," *Computers & Electrical Engineering*, vol. 86, Article No. 106719, 2020.

[5] Lei Zhang, Mingxing Luo, Jiangtao Li, Man Ho Au, Kim-Kwang Raymond Choo, Tong Chen, and Shengwei Tian, "Blockchain based secure data sharing system for internet of vehicles: A position paper," *Vehicular Communications*, vol. 16, pp. 85–93, 2019.

[6] Rakesh Shrestha, Rojeena Bajracharya, Anish P Shrestha, and Seung Yeob Nam. A new-type of blockchain for secure message exchange in vanet. *Digital Communications and Networks*, pages 1–14, 2019.

[7] Jiawen Kang, Zehui Xiong, Dusit Niyato, Dongdong Ye, Dong In Kim, and Jun Zhao, "Toward secure blockchain-enabled internet of vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2906–2920, 2019.

[8] Zhe Yang, Kan Yang, Lei Lei, Kan Zheng, and Victor CM Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495–1505, 2018.

[9] Xiaohong Zhang and Xiaofeng Chen, "Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network," *IEEE Access*, vol. 7, pp. 58241–58254, 2019.

[10] Yao-Tsung Yang, Li-Der Chou, Chia-Wei Tseng, Fan-Hsun Tseng, and Chien-Chang Liu, "Blockchain-based traffic event validation and trust verification for vanets," *IEEE Access*, vol. 7, pp. 30868–30877, 2019.

[11] Farah Kandah, Brennan Huber, Anthony Skjellum, and Amani Altarawneh. A blockchain-based trust management approach for connected autonomous vehicles in smart cities. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0544–0549. IEEE, 2019.

[12] Jiawen Kang, Rong Yu, Xumin Huang, Maoqiang Wu, Sabita Maharjan, Shengli Xie, and Yan Zhang, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, pp. 4660–4670, 2018.

[13] Lun Li, Jiqiang Liu, Lichen Cheng, Shuo Qiu, Wei Wang, Xiangliang Zhang, and Zonghua Zhang, "Creditcoin: A privacy-preserving blockchain-based incentive announcement network for communications

of smart vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 7, pp. 2204–2220, 2018.

[14] Yunlong Lu, Xiaohong Huang, Ke Zhang, Sabita Maharjan, and Yan Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4298–4311, 2020.

[15] Rakesh Shrestha, Rojeena Bajracharya, and Seung Yeob Nam. Blockchain-based message dissemination in vanet. In *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, pages 161–166. IEEE, 2018.

[16] Zhaojun Lu, Wenchao Liu, Qian Wang, Gang Qu, and Zhenglin Liu, "A privacy-preserving trust model based on blockchain for vanets," *IEEE Access*, vol. 6, pp. 45655–45664, 2018.

[17] Zhaojun Lu, Qian Wang, Gang Qu, and Zhenglin Liu. Bars: a blockchain-based anonymous reputation system for trust management in vanets. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 98–103. IEEE, 2018.

[18] Bao-Kun Zheng, Lie-Huang Zhu, Meng Shen, Feng Gao, Chuan Zhang, Yan-Dong Li, and Jing Yang, "Scalable and privacy-preserving data sharing based on blockchain," *Journal of Computer Science and Technology*, vol. 33, no. 3, pp. 557–567, 2018.

[19] Xumin Huang, Dongdong Ye, Rong Yu, and Lei Shu, "Securing parked vehicle assisted fog computing with blockchain and optimal smart contract design," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 426–441, 2020.

[20] Muhammad Umar Javed, Mubariz Rehman, Nadeem Javaid, Abdulaziz Aldegheishem, Nabil Alrajeh, and Muhammad Tahir, "Blockchain-based secure data storage for distributed vehicular networks," *Applied Sciences*, vol. 10, no. 6, Article No. 2011, 2020.

[21] Kexin Shi, Liehuang Zhu, Can Zhang, Lei Xu, and Feng Gao. Blockchain-based multimedia sharing in vehicular social networks with privacy protection. *Multimedia Tools and Applications*, pages 1–21, 2020.

Sanjeev Kumar Dwivedi is pursuing a Ph.D. in the Department of CSE from Dr. Shyama Prasad Mukherjee International Institute of Information Technology Naya Raipur, (DSPM-IIITNR), Chhattisgarh,India. He received an M.Tech Degree in the Department of CSE from Pondicherry University, Puducherry, India, in 2013. His research interest includes VANET security, cryptography, and blockchain.

Ruhul Amin received a Ph.D. in CSE from the Indian Institute of Technology (ISM) Dhanbad, Jharkhand, India, in 2017. Presently, he is working as an Assistant Professor in the Department of CSE, Dr. Shyama Prasad Mukherjee International Institute of Information Technology Naya Raipur (DSPM-IIITNR), Chhattisgarh, India. His research interest includes VANET security, authentication protocol, and blockchain.

Satyanarayana Vollala received a Ph.D. in CSE from National Institute of Technology, Tiruchirappalli, (NITT) Tamilnadu, India, in 2017. Presently, he is working as an Assistant Professor in the Department of CSE, Dr. Shyama Prasad Mukherjee International Institute of Information Technology Naya Raipur (DSPM-IIITNR), Chhattisgarh, India. His research interest includes Hardware security and theoretical computer science.