University of Nevada, Reno

Blockchain Based Decentralized Applications & Trust Management for VANETs

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in
Computer Science and Engineering

by

Sowmya Kudva

Shahriar Badsha, Ph.D. - Thesis Advisor
Shamik Sengupta, Ph.D. - Thesis Co-Advisor
May 2021

THE GRADUATE SCHOOL

We recommend that the thesis
prepared under our supervision by

SOWMYA KUDVA

entitled

Blockchain Based Decentralized Applications & Trust Management for VANETs

be accepted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

Shahriar Badsha, Ph.D
Advisor

Shamik Sengupta, Ph.D
Co-advisor

Hanif Livani, Ph.D
Graduate School Representative

David W. Zeh, Ph.D., Dean
Graduate School

May, 2021

Abstract

Decentralized vehicular Ad-hoc Networks (VANETs), a promising technology to improve the Intelligent Transportation System (ITSs), face severe lagging in actual deployment and its extensive usage due to major unresolved issues such as security, data reliability, user privacy, and safe routing protocols.

To overcome these issues, there is an urge to identify a platform that best suits VANET's easy deployment and usage in a decentralized fashion. In this regard, blockchain has received much attention as an emerging technology to provide better security on data sharing among many participants without an intermediary. This thesis aims to investigate blockchain technology's capability to secure vehicular data and vehicular node trust scores over a tamper-proof decentralized ledger that guarantees security, immutability, and accountability in Peer-to-Peer (P2P) networks such as VANET.

Firstly, we explore how to leverage blockchain technology to design a specific application in the domain of decentralized VANETs, such as ride-sharing. We analyze the decentralized architecture for this application using smart contracts, and through experiments, we evaluate the costs associated with it. This framework serves as a basis for our further study to solve more challenging research problems in the consensus algorithm. Choice of a consensus algorithm directly affects the performance [1] of a blockchain-based system in terms of transaction confirmation delays. In a VANET based on blockchain, the Proof of Work (PoW) and Proof of Stake (PoS) consensus might not be the best selection due to resource constraints and unfairness, respectively. In an attempt to improve consensus in a VANET application based on blockchain, we present the design of a novel consensus mechanism named Proof Of Driving for our previously presented ride-sharing application. We demonstrated that

POD clubbed with a real-time service standard score protocol efficiently optimizes the number of miner nodes. The extensive experimental and security analyses presented on proposed consensus and service standard protocols demonstrate the effectiveness, security, and feasibility of miner node selection.

However, VANET is not secure as vehicular communication is critically vulnerable to several kinds of active and passive routing protocol attacks. The most severe attack in routing is the Black Hole attack, which deteriorates the network's performance by dropping or misusing the intercepted data packets without forwarding them to the correct destination. This greatly hinders the application availability. Hence in the final chapter of this thesis, we experiment by incorporating trust models in VANET routing protocols to achieve a more efficient packet forwarding process. The results showed an improved packet delivery ratio and throughput of the entire network. The trust model should be able to resist various attacks and preserve the privacy of vehicles simultaneously. Hence we presented how to leverage consortium blockchain to secure vehicles' trust scores and distribute node trust in a decentralized network more efficiently. We evaluated the trust score aggregation process by the authorized RSUs, the time consumed for consensus, and updated trust score distribution. The results showed that the blockchain-based trust management provides an effective trust model for VANETs with transparency, conditional anonymity, efficiency, and robustness while efficiently eliminates the black hole nodes.

Acknowledgments

I would like to extend sincere thanks to my advisor, Dr. Shahriar Badsha, who is a constant source of inspiration and enlightenment required for all the phases of my research. I also thank my co-advisor, Dr.Shamik Sengupta, and committee member Dr. Hanif Livani for taking the time to review this thesis.

Finally, I thank my family for being pillars of emotional support and constant motivation throughout my time in this school. I also thank my friends for their support.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Vehicular ad hoc Networks (VANETs) have emerged mainly to enhance road safety, traffic efficiency, and passenger comfort. While intelligent transportation system (ITSs) focuses on providing intelligence to the roadside and vehicles' systems, VANET focuses on providing communication between those systems.

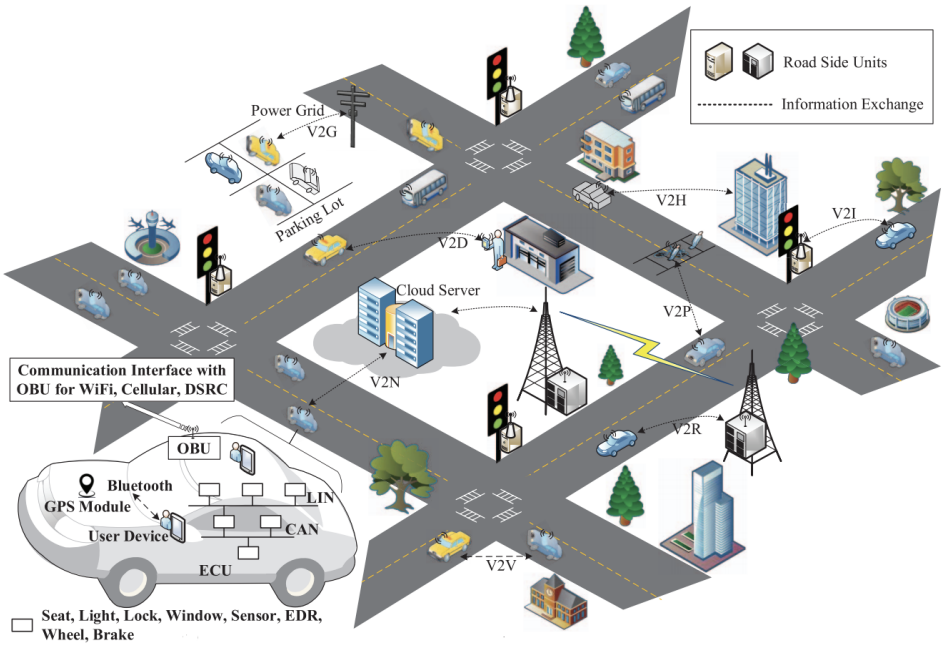


Figure 1.1: V2X Technology used in communication

[1]

VANET supports three or more architectures as shown in Fig 1.1. Vehicles that are equipped with sensors, actuators, and computation capabilities communicate directly with each other (V2V) or with roadside infrastructure (V2I) in an ad-hoc manner over an open wireless communication mode. Vehicles can also communicate with pedestrians (V2P) to support different types of applications. In Hybrid architecture, all the communication modes (V2V, V2I & V2P) are combined to achieve effective communication.

The message exchange capabilities of V2V, V2I, and V2P can be used to develop various ad-hoc VANET applications. It would allow direct, instant, and flexible communication between vehicles and roadside passengers, eliminating the third party. However, an ad-hoc network also brings risks and challenges with it, as it is easy for an attacker to eavesdrop, tamper with, or forge the information shared by users (vehicles/pedestrians) on an open wireless communication network.

1.1 Motivation

It is of vital importance to design a secure, data traceable, and efficient platform yet decentralized in nature. In this direction, blockchain, as an emerging technology with the features of security, credibility, tamper resistance, and traceability, has set off a research boom all over the world. Various initiatives have been recently launched to investigate the blockchain technology's capability in enabling vehicular application over a tamper-proof decentralized ledger [3,4]. For our study, we consider ride-sharing (carpooling) VANET applications that are increasingly becoming the most attention-grabbing trends in the recent years [5,6]. However, it is equally important to carefully

[1]Fig 1.1 is added from a survey paper [2]

choose a consensus algorithm for a blockchain-based VANET application as it directly affects the performance [4] in terms of transaction confirmation delays.

In a public blockchain application, anyone can join or leave the network freely, resulting in all nodes having read and write permissions for data. It is an arduous task for VANET applications based on the public blockchain to reach a consensus quickly among the nodes. The classic Proof of Work (PoW) [7] and the Proof of Stake (PoS) might not be the best way to contribute to the vehicular network due to its focus on high computation power and stake-based selection, respectively. This motivated us to work towards building a proof variant that is adaptable to the nature of the vehicular nodes (mobility and resource constraint devices), that is more randomized, impartial, and dynamic such that only a smaller number of randomly selected vehicles get to perform the mining in a ride-sharing application based on a public blockchain.

However, VANET is still not fully secure as they are exposed to increased risk of being a target for various types of internal attacks. Hence the system interfaces and routing protocols have vulnerabilities that hinder the application availability. An internal malicious node intercepts the passing through packets and drops them, using the routing protocol's vulnerability. This attack is known as a black hole attack [8]. To achieve a more efficient packet forwarding process and to mitigate packet drop attacks in VANETs, trust-based solutions help to detect selfish nodes that act as black holes in the network. It is critical and challenging to develop a decentralized, reliable, and consistent trust management system in vehicular networks. Due to the features of decentralization, consistency, and tamper-proofing, blockchain can be a promising technique to also help cope with the trust management problems in vehicular networks.

1.2 Contribution

Motivated by the above-discussed challenges, we present our contribution as follows. In Chapter 4 of this thesis, we presented a system **PEBERS: Practical Ethereum Blockchain-based Efficient Ride-Sharing Service**, in which we demonstrated how a decentralized ride-sharing system based on the blockchain could be developed to keep track of ride data. In this context, we explored smart contracts to build and deploy various functionalities for the ride-sharing application. Our experiments showed that the smart contracts proposed consumed less Gas and proved that it is an efficient system than other existing centralized systems concerning passengers' expenses and profitability.

In regards to the discussion around a need for a new consensus mechanism for a vehicular network based on blockchain, in Chapter 5 of this thesis, we explored solutions to the problem of (1) how can we identify criteria to optimize the number of vehicular nodes performing mining in VANET application and (2) how randomness can be introduced in the model for miner node selection yet without compromising the quality of miners for an application considered in chapter 4. We introduced a new proof-variant named Proof of Driving (PoD) that brings randomness in a blockchain-based public ride-sharing VANET application. We further narrowed down the miner nodes by considering the Service Standard Score, which indicates various performance factors such as error rate, block creation, and reputation. Our experiments evaluated the new mechanism concerning computation cost and time, and fairness to demonstrate the effectiveness.

To design an efficient trust management system to address the blackhole attacks in VANET, in chapter 6, we proposed a decentralized trust management system that stores and distributes the trust scores of vehicular nodes. We introduced a consortium blockchain-based trust score system as a solution to detect and blacklist

multiple blackhole nodes from the network, much different from conventional methods of elimination. We evaluated the proposed system concerning computation cost and time to aggregate trust scores and disseminated it in the network so that malicious nodes can be blacklisted, resulting in better network metrics.

The main contributions of this thesis are summarized as the following;

- In chapter 4, we perform a case study by designing a ride-sharing application based on a public blockchain. We built and deployed a driver smart contract prototype and evaluated them by deploying them to a local public network named Ganache.

- In chapter 5, we have designed and developed a new proof variant named proof-of-driving, alongside service standard score for selecting miner nodes for an earlier introduced vehicular network application (ride-sharing) based on blockchain.

- Finally, in chapter 6, we have presented a blockchain-based trust management system for a vehicular network that can be used to store and disseminate trust scores and showed through simulation how the network metrics such as throughput rate and packet forward ratio could be improved by blacklisting malicious nodes.

The next chapter will briefly discuss background information concerning types of blockchain networks and various consensus algorithms.

Chapter 2

Background

This chapter defines 'Blockchain Technology' in a nutshell and its network types, followed by a brief introduction to various consensus mechanisms in VANET and its limitations. We also give a short introduction about smart contracts, Solidity, and the Ethereum platform.

2.1 Blockchain Technology

The original premise of blockchain [9] is to establish a peer-to-peer (P2P) network for monetary value transfer such as Bitcoin. In this network, no financial institution is required to make a value transfer. A blockchain is essentially a chain of hashed 'blocks' where each block contains a time-stamp, the previous block's hash, and a collection of transactions, as illustrated in Fig. 2.1. A set of unconfirmed transactions are bundled together as a block. Peers in the network use their processing power to solve the mathematical puzzle to validate transactions in the block without any centralized administration. Once the block is validated, it is added to the blockchain, and the transaction gets visible to the entire network and thus becomes immutable.

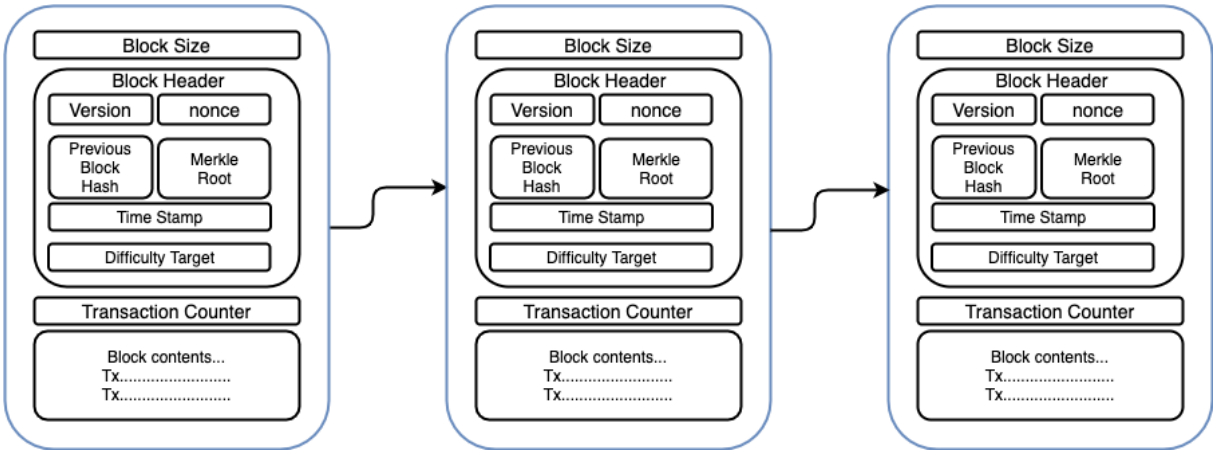


Figure 2.1: Simplified Chain of Hashed Blocks

There are different various types of blockchains, and each incorporates a different level of privacy and security for different use cases. Below is a description of the different types of blockchains as described by authors of [10]

Public Blockchain: In a public blockchain, all miners participate in the consensus determination process, and therefore the ledger is totally visible to all or any participants. Public blockchains are permissionless and do not implement access control regarding transaction acceptance.

Private Blockchain: Private blockchains are based on a centralized architecture where one business or entity controls all of the nodes in the blockchain and writes and validates all transactions [10]. This allows higher efficiency and strict permissions on who can participate in the network. However, all of the downsides that accompany centralized architecture still remain.

Consortium Blockchain: In a consortium blockchain, only trusted nodes could participate in the validation of blocks, but these trusted nodes are not defined to a single organization or entity. This can provide some of the benefits of a private blockchain, such as efficiency and privacy of transactions without compromising the public blockchain's decentralized nature.

2.2 Smart contracts, Solidity & Ethereum

Smart contracts: They can be defined as contractual constraints deployed on the blockchain that act as an immutable agreement and can receive or execute transactions. Smart contracts are developed using a scripting language called Solidity, which is compiled into EVM byte code and then deployed on the blockchain.

Ethereum Blockchain: The Ethereum blockchain is a consortium blockchain network that enables people to develop and deploy their decentralized applications via smart contracts. Users can create an Ethereum account that is assigned with an address. Every computational step of transaction made has an associated gas price.

2.3 Consensus Mechanisms

Consensus in a blockchain is a process where all the network peers reach a joint agreement about the present state of the distributed ledger. At present, the most common consensus algorithms are Proof of Work(PoW), Proof of Stake(PoS), and Practical Byzantine Fault Tolerance(PBFT). From the emergence of Bitcoin to today, there are more than 30 consensus algorithms [11], most of which are based on the above three consensus algorithms.

Proof of Work (PoW): PoW consensus protocol, which was introduced via Bitcoin, requires each validating user to prove that she has performed a computational action by solving complex cryptographic problems using their computational resources. Hence PoW is computation-intensive as each miner must deliver a hash querying rate as high as possible to win the puzzle-solving race [7]. Hence this scheme is unsuitable for the vehicular network due to resource limitations in vehicles.

Proof of Stake (PoS): PoS [12] completely replaces the mining operation with an alternate approach involving a user's stake or ownership of virtual currency in the blockchain system. The cryptographic calculations in PoS are much more straightforward for computers to solve. Nevertheless, a rich validator may keep on winning the bid resulting in undesirable centralization around those prominent stakeholders and raise significant trust concerns [13]

Practical Byzantine Fault Tolerance (PBFT): PBFT [14], as opposed to the other discussed consensus mechanisms, was not explicitly introduced for blockchain. This consensus model gets the idea from the byzantine generals' problem in which all generals' votes need to be transmitted to all others through broadcasting. It works efficiently even when a portion of the network is faulty. However, PBFT only scales to few tens of nodes since it must exchange messages to succeed in consensus on one operation among n servers leading to $O(n^2)$ complexity. Hence, to be efficient in a VANET application such as ride-sharing, the number of nodes performing consensus in the network should be significantly reduced.

Chapter 3

Related Works

In this chapter, we have discussed a few of the published schemes related to the different areas of our thesis.

3.1 Blockchain based ride-sharing in VANET

Several blockchain-based ride-sharing applications have arisen thus far to bring changes into the operation of online carpooling systems with innovative ideas. Some companies developed a blockchain-based ride-sharing platform, e.g., DACSEE and Arcade City, termed Uber 2.0. Each of these efforts mentioned below varies mainly based on their purpose and implementation.

In [15] real-time ride-sharing has been proposed to maximise the revenue of the platform without compromising the the standard. Their results demonstrate the effectiveness and efficiency of their framework. This work received much attention.

A scheme, called GreenRide, is proposed in [16] which illustrates the integration of the blockchain into rideshare application to incentivize users to share their rides with colleagues and hence decrease carbon emission. A decentralized ride-sharing scheme

with reputation is proposed in [17]

In [18], a scheme called Oride is proposed to enhance accountability and improve privacy by encrypted communication with the Service Provider. However, calculations involving encryption might cause latency within the system.

Our work is primarily an extension of the work described here. However, in our work, driver contracts are deployed individually. Our focus is on building a public blockchain implementation for a ride-sharing platform as a preliminary for our future work. Hence, we investigated the variation of transaction and gas cost for creating, building and deploying smart contracts functionalities associated with the car-sharing system. We study specific issues such as consensus associated with a public blockchain in VANET and propose solutions in future chapters.

3.2 Consensus for Blockchain-based VANET

With the continual development of blockchain technology, consensus algorithms have gradually become a new research hotspot [19]. Much relevant work has already been done in this regard. In [20] authors designed their consensus phases based on the Byzantine Fault Tolerates algorithm while proposing a privacy-preserving incentive announcement network based on the blockchain via an efficient anonymous vehicular announcement aggregation protocol.

Authors of [21] indicated that Delegated PoS (DPoS) is particularly suitable for vehicles to establish blockchain-based vehicular networks in ITS ecosystems. In [22] a blockchain-based distributed framework for the automotive industry in the smart city is proposed that includes a novel miner node selection strategy based on a fruit fly algorithm with an underlying PoW consensus mechanism. Kang et al. proposed an enhanced DPoS consensus scheme with a two-stage soft security solution for se-

cure vehicle data sharing and ensured secure miner selection based on reputation to establish blockchain-enabled Internet of Vehicles (IoV) [23].

Work in [24] studied the problem of mining pool selection in a blockchain network adopting the proof of work scheme, and authors modeled the dynamics of pool selection among individual miners as an evolutionary game. Even in [25], authors have focused on the consensus propagation delay problem in PoS based consortium blockchain networks. They formulated a Stackelberg game to trade off security requirements, verification delay, and cost, which jointly maximizes the blockchain user's utility and the miners. However, many miners participating in block generation result in more significant verification delays and higher costs.

Several other works have chosen many different approaches. In [26], Peterson et al. proposed a random miner selection consensus protocol to elect a miner. [27] proof of reputation is proposed in which reputation-based weighted voting is performed as an alternative way to provide a solid deterministic consensus in a permissionless distributed blockchain system.

In [28] authors have proposed a private vehicular blockchain called Parkingchain, where the parked vehicles (PVs) can share their idle computational resources with service requesters (SRs). Authors of [29] present a new technique called Algorand that uses a new Byzantine Agreement (BA) protocol to reach consensus among users on the next set of transactions.

Chapter 5 of this thesis employs an approach consisting of proof variant PoD and filtering technique S_c to efficiently choose nodes for adopting PBFT to make it scale in a more extensive network of vehicles.

3.3 Trust Management based on Blockchain

In [30], the authors proposed a combination of greedy geographic routing protocol and trust tables to include a reliability parameter for vehicular nodes. However, their solution is suitable for only a low-density network where the car moves in straight lines. In [31] authors built an intelligent intrusion detection system (IDS) based blackhole detection system for VANETs. It can detect novel attacks (variations of blackhole attacks) as well. However, it is time-consuming and requires more computational resources.

Table 3.1: Comparative Analysis of Various Approaches of the Existing Trust Models For VANET

Approaches	[32]	[30]	[31]	[33]	[34]	[13]	[35]	OurWork
Decentralized Trust Management	✓			✓	✓	✓	✓	✓
Centralized Trust Tables		✓						
Detect Blackhole Greyhole attack			✓	✓			✓	✓
Detect Multiple Blackhole nodes								✓
Based on Entity Trust	✓							✓
Reputation on Messages Exchanged					✓	✓		
Leverage Blockchain	✓					✓	✓	✓

Several solutions were proposed based on trust value such as in [33], every node maintains a trust table in addition to the routing table in which every individual node stores the trust value of its one-hop neighboring nodes by monitoring in promiscuous mode to observe the forwarding and dropping of packets by the neighboring nodes.

They also implement a trust manager that sends and receives trust tables to/from RSUs. However, there is a high potential for a single point of failure with this design. In [34], authors have proposed a technique incorporating a trust table for holding the honest nodes. The RREP is overloaded with an extra field that indicates the reliability of the replying node.

As we have seen from the above discussion, most of the proposed methods brought some novelty to the attack detection scheme. Besides, they suffered from drawbacks such as scalability [30], computational overhead issues on intermediate, and source node [31]. Besides, a trust-based system like [33] fails in the case of an extensive network.

Based on blockchain's decentralization nature, trust management issues can effectively be conducted using RSU's on the blockchain network in VANETs. Authors of [32] explore a blockchain-based anonymous reputation system to establish distributed trust management in VANETs. Similarly, the authors of [13] proposed a blockchain-based decentralized trust management system in vehicular networks using the Bayesian inference model for the received messages to assess its credibilities. In [35], authors developed a countermeasure for blackhole attacks in VANETs consisting of detection, identification, and prevention of blackhole nodes using a backtracking algorithm.

In the next chapter, we detail the case study of carpooling applications in VANET based on blockchain and present the results regarding cost, efficiency, and other benefits by using smart contracts.

Chapter 4

PEBERS: Practical Ethereum Blockchain based Efficient Ride Sharing Service

In this chapter, we have specifically considered the ride-sharing application in the domain of VANET and explore how blockchain technology can be leveraged to design it in a decentralized architecture using smart contracts. Our focus is mainly on evaluating the costs associated with it, as this framework serves as a basis for our future chapters to solve more challenging research problems.

4.1 Motivation

Centralized ride-sharing system architecture demands high maintenance to serve at a large scale, making it much costlier and risky. It is more prone to information monopolies or information islands. Last but not least, existing ride-sharing schemes

manage the payments, giving no control to the users, have high computational costs and communication overheads.

A simple, robust system that is cost-effective, involving direct communication between vehicles and passengers (clients) using the vehicular ad-hoc network would provide more control to the users concerning payments. However, the decentralized design also presents unique challenges, such as the lack of trust between users and decentralized infrastructure oversight. The vehicle or the passengers' data needs to be shared securely and stored in a decentralized manner. Additionally, the records of ride-sharing should not be disclosed, altered, or may be deleted.

4.2 Contribution

Blockchain has gained exceptional recognition while addressing the issues mentioned above because of its decentralized data auditability, anonymity, and immutability. Hence, we have explored the possibility of implementing ride-sharing services in a decentralized fashion based on blockchain technology, which allows us to deviate from centralized platforms to decentralized ones. The main contributions of this paper are summarized below.

- We propose the design of a system PEBERS: Practical Ethereum Blockchain-based Efficient Ride-Sharing Service, in which we demonstrate how a decentralized system based on the public blockchain can be developed to keep track of ride data
- We built a prototype of smart contract and evaluated them by deploying to a local network to calculate the expenses incurred for both drivers and passengers by employing such a design.

- To simulate using the real EVN, a sensible gas price was used for every smart contract call. Numerical results are provided in our figures and tables corresponding to gas consumption and transaction cost.

4.3 Proposed System Model

PEBERS system model is made of three layers. At the user's layer, we have all of the end-user entities (drivers and passengers), as shown in Fig 4.1. In the infrastructure layer, we have Road Side Units (RSU) with computing and storage capacities distributed in a decentralized manner as Registration Authority. Each of these nodes is maintaining a replica of the transaction ledger. Each of the entity's responsibilities is explained as follows:

Registration Authority: RA may be a legitimate and authoritative component of a blockchain system. To maintain secure communication between every vehicle and passenger node, they are made to register with the Register Authority (RA). Road side units (RSU) are equipped with computation, communication, and network capabilities. We use a set of RSU as a few semi-trusted nodes in the network. It can be configured in a decentralized manner to work as RA.

It is the aptest method in current research work to possess permission granting authority like RA. In [36] System Administrator is considered into the design for registration. In our work, we assume the existence of a RA that administers registration to the blockchain network. RA has an important unit called key generating unit that generates a unique passphrase for passengers and drivers. From this, keys are generated, as shown in Fig 2. When an entity registers to the network, its identity is verified and tracked by RA.

Vehicular Nodes : In our blockchain environment, each vehicular node is rep-

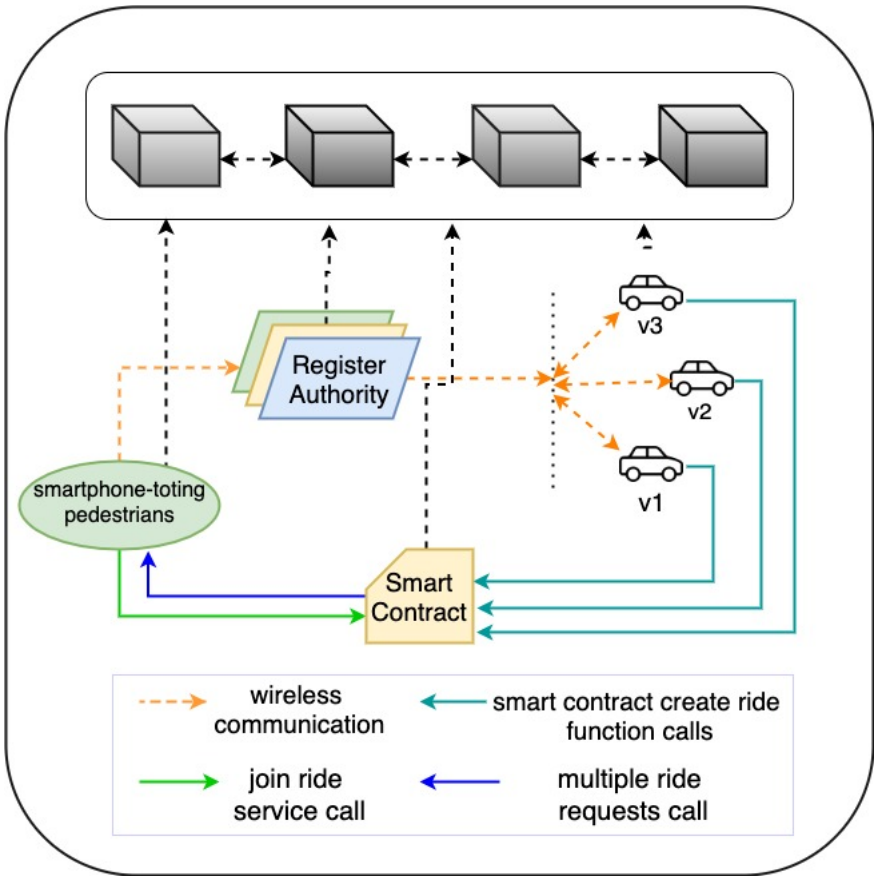


Figure 4.1: Decentralized, Blockchain based Ride-Sharing System architecture

resented as V_i where $i \in \{1, 2, 3, ..., N\}$ and N is the total number of vehicular nodes in the network. Vehicles are installed with advanced communication devices, wireless transmission modules like IEEE802.11, GPS receivers and can perform simple computations such as calculating location coordinates, determining their location by matching the electronic maps, and verifying transactions.

Passenger Devices: Passengers also need necessary hardware equipment to support the communication. Any passenger device is denoted by P_j where $j \in \{1, 2, 3, 4...m\}$ and m is the total number of passenger nodes in the network. They are equipped with wireless pocket devices to call a vehicle for a ride, which can be regarded as a cheap device with an electronic map and simple input/output. These devices are termed light nodes as they only participate in obtaining services and leaving service ratings.

Transactions: Transactions are the primary communication primitive in blockchain for information exchange among entities in the system model. In our model, the transaction is the record of every event such as passenger requesting for the ride, driving vehicle accepting the request with a response, passenger acknowledgment at a particular date and time, along with the amount of fee paid.

4.4 Methodology

- **Registration phase**: Registration occurs at the system users layer, which encompasses all the different entities of the ride-hailing application, including drivers and passengers. Firstly, users sign-up for the application. This phase is executed only once in the process. Users register with the Registration Authority(RA), as shown in Fig 4.2.

  After passing the identity authentication by a RA, a legitimate driver d_i obtains

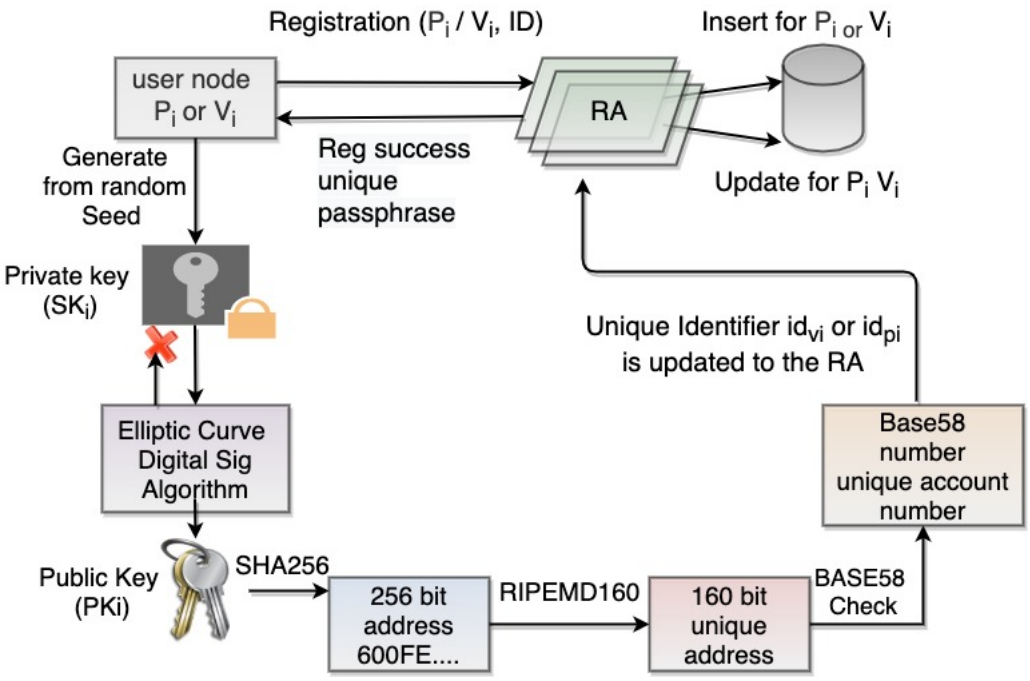


Figure 4.2: Key Generation Process

its unique passphrase. d_i derives secret key SK_{di}, public key PK_{di} and a unique id id_{di} from it. It can be represented as below:

$$d_i \rightarrow \{\ id_{di},\ SK_{di},\ PK_{di}\ \}$$

Passengers also register using the mobile devices through a secure protocol. A legitimate passenger p_i receives his own passphrase in the similar manner and derives SK_{pi} , PK_{pi} keys and identity id_{pi} from it. It can be represented as below:

$$p_i \rightarrow \{\ id_{pi},\ SK_{pi},\ PK_{pi}\ \}$$

After users have been registered and credentialed, they can use their digital identities ie id_{pi}, Id_{di} to access the services.

- **Deploying driver smart contract**: A driver is uniquely identified by his id_{di} in the network. The registered driver deploys the smart contract on the network

and deposits one ether initially on the smart contract storage. Algorithm 1 represents our driver smart contract [37]

- **Ride Requesting phase**: In this phase, passenger notifies to the nearest RSU node about its ride requirement. A registered passenger p_i looking for a ride, holding an identity id_{pi} and a reputation value rep_{pi} creates a request REQ_{pi} with her current location O_{pi}. This request is received by the nearest RSU node (within the communication range).

- **Forwarding ride request by RSU node** When a ride request is received by a RSU, it tries to broadcast it to many registered driver vehicles nearby via V2I communication range. Interested drivers within the desired region of passenger, first checks the REQ_{pi} from passenger and sends *createRide* request to passengers by specifying their price per mile quote via an application interface.

- **Accepting ride**: If the passenger p_i does not want to accept anyone from the list of waiting drivers, ride request should be time out after certain period. Otherwise, id_{pi} sends *joinRide* in response to the chosen driver d_i. By doing so, passenger is in formal agreement to ride with the selected driver.

- **Pick Up**: On pickup, the driver calls $DriverConfirmed$ function of the smart contract to initiate the ride and changes the status of the vehicular node as $driverConfirmed$. On ride completion, passenger makes a call to *arrived* method of smart contract which transfers the cost of ride to driver account automatically. Users leave a rating to each other which gets accumulated in the reputation parameter.

- **Ride Cancellation**: If a passenger or driver decides to drop off from the ride they can call a *cancelRide* function of the smart contract. However if the ride is cancelled after selecting the driver, some amount of ethers from the deposit

Algorithm 1: Algorithm for *RideShare* contract

```
Struct Passenger contains
    price & state;
end
Struct Ride contains
    address driver; drivingCost; capacity; confirmedAt;
    originAddress; destAddress; address payable[] passengerAccts ;
end
rides → array of type Ride;
rideCount → length of rides array;
if function() == createRide then
    read the input params → obj of Ride
    create empty passengerAccounts array;
    if exists(all params of Ride) then
        rides.push(obj(Ride);
    end
end
if function() == joinRide then
    read the input params → rideNumber
    fetch the curRide = rides[rideNumber];
    if exists(msg.value == curRide.drivingCost) then
        passenger = msg.sender;
        passenger.price = msg.value;
        passenger.state = "initial";
        curRide.passengerAccounts.push(passenger);
    end
end
if function() == confirmDriverMet then
    read the input params → rideNumber
    fetch the curRide = rides[rideNumber]
    if exists(curRide.passenger == msg.sender) then
        set curRide.passengers.msg.sender.state = "driverConfirmed";
    end
end
if function() == arrived then
    curRide.driver.transfer(curRide.passengers[msg.sender].price);
    curRide.passengers[msg.sender].state = "completion";
end
if function() == cancelRide then
    if (curTime ≤ curRide.confirmedAt) then
        if (msg.sender == driver) then
            curRide.passenger.transfer(curRide.DriverCost);
        end
        if msg.sender == passenger then
            msg.sender.transfer(curRide.passengers.msg.sender.price);
        end
    end
end
```

is transferred to the driver account to respect their time. Similarly, if the driver is cancelling the ride for some reason, ethers from the initial deposit will be transferred to passenger's account.

4.5 Experimental Setup & Results

In this section of the paper, we evaluate the performance metrics of our smart contracts. Performance refers to how inexpensive we can make our smart contract's code while keeping the functionality. The smart contracts' actual execution cost refers to the amount of gas that is used by the opcode of the operations in smart contracts. For testing the gas consumed, we used Ganache, a local blockchain network to deploy, and Truffle as our development environment to compile the smart contracts. The smart contracts were written in a programming language called Solidity which is provided by default by the EVN platform. Gas is the execution fee for every operation on the EVN, and Ether is a digital currency, which is the unit of gas. GWEI is the smallest unit of gas. 1 ether is equal to the 1,000,000,000 gwei. We first start our experiment by deploying the Ride share smart contracts.

4.5.1 Analysis on cost of deploying the RideShare and Authentication smart contract

The *RideShare* smart contract as shown in the pseudo-code contains several functions such as createRide, joinRide, cancelRideand, and variables as explained in the earlier section. The *Authentication* smart contract stores a driver and passenger's name, as well as ID. It is only used once to authenticate the user into the system. A mapping variable "Users," stores a User to a specific address as well. The smart contract checks

if the user exists by seeing if id_{di} and id_{ri} are valid ID's that start with "0x". The smart contract checks that the User ID cannot be equal to "0x0" as well because this ID is null. The Contract's next cost is when the valid User ID is stored into the mapping variable "Users".

Table.4.1 displays the estimated cost of driver for deploying the *rideshare* and *authentication* smart contract on the local ethereum network. These costs are one-time during the setup. We also measure the transaction cost associated with the create ride function call. This cost of *createRide* is for each rideshare trip. Gas price is fixed at 125 GWEI per unit of gas consumed and ether price at 215.08 USD as of September 20th, 2019. Completing 20 trips, the driver spends about 10 USD.

The transaction costs from Table 4.1 were tested in the Remix IDE using solidity version 0.4.25. The transaction and execution cost for each function call was analyzed in Remix IDE's debugger.

Table 4.1: Smart Contract's Transaction and Execution cost

Smart Contracts $F(x)$	**execution cost** (Gas)	**cost in ethers** (Gas)
RideShare	724233	0.0905291
Authentication	164081	0.0205101
CreateRide	156828	0.0196035

4.6 Summary

In this chapter saw how blockchain technology for ride-sharing services combined with smart contracts could be implemented. The crucial feature is our smart contract deployed in the network, which helps passengers join the rides. Furthermore, we showed how an automatic deposit transfer from one account to other brings trust in the system. However, we did not emphasize the problem associated with consensus in

the network. In the next chapter, we specifically discuss the drawbacks of consensus mechanisms in a public blockchain and propose a solution.

Chapter 5

Towards Secure and Practical Consensus for Blockchain based VANET

Previously, we explored smart contracts to make a ride-sharing application based on trust-less decentralized architecture. That was tested in the local ethereum test network, which uses the default consensus algorithm (PoW). In this chapter, we focus our study more on the performance of the blockchain system [4] in terms of transaction confirmation delays (for the same ride-sharing application), which is directly affected by choice of the consensus algorithm. We present a theoretic design for a new consensus mechanism and execute tests to evaluate its efficiency in this direction.

5.1 Motivation

For the ride-sharing application, the classic Proof of Work (PoW) [7] might not be the best way to contribute to the vehicular network due to its focus on high computation power. On the other hand, the Proof of Stake (PoS) may only incorporate stake-based selection, which can give rise to unfair conditions. This motivates us to find a proof variant that is adaptable to the nature of the vehicular nodes (mobility and resource constraint devices), more randomized, and finally impartial and dynamic such that only a smaller number of randomly selected vehicles get to perform the mining. Hence, we focus on exploring solutions to the optimization of the number of vehicular nodes performing mining in VANET application and how randomness can be introduced in the model for miner node selection yet without compromising the standard of miners [1].

5.2 Contribution

The main contributions of this research are summarized below.

- We introduce a new proof variant named Proof of Driving (PoD) into the design of blockchain-based ride-sharing service in VANET, which consumes fewer resources than PoW, maintains fair selection as well as the randomness of consensus nodes in a public distributed network of vehicles.

- We design a real-time service standard score protocol S_c to efficiently optimize the number of miner nodes considering their past performance and eliminate the poor quality or malicious nodes from being part of the consensus.

- We present extensive experimental and security analyses on proposed PoD and S_c protocols to show the effectiveness, security, and feasibility of miner node

selection.

5.3 Components of Proposed System Model

As seen in the previous chapter system design, a decentralized network in our model mainly includes several vehicular nodes, passenger handheld devices that communicate with each other by sharing information via DSRC radio, and Register Authority (RA). Fig 5.1. illustrates how different components of the system are connected. We have enhanced the design with some additional sub-components for RA. Detailed descriptions of these components are given in the following.

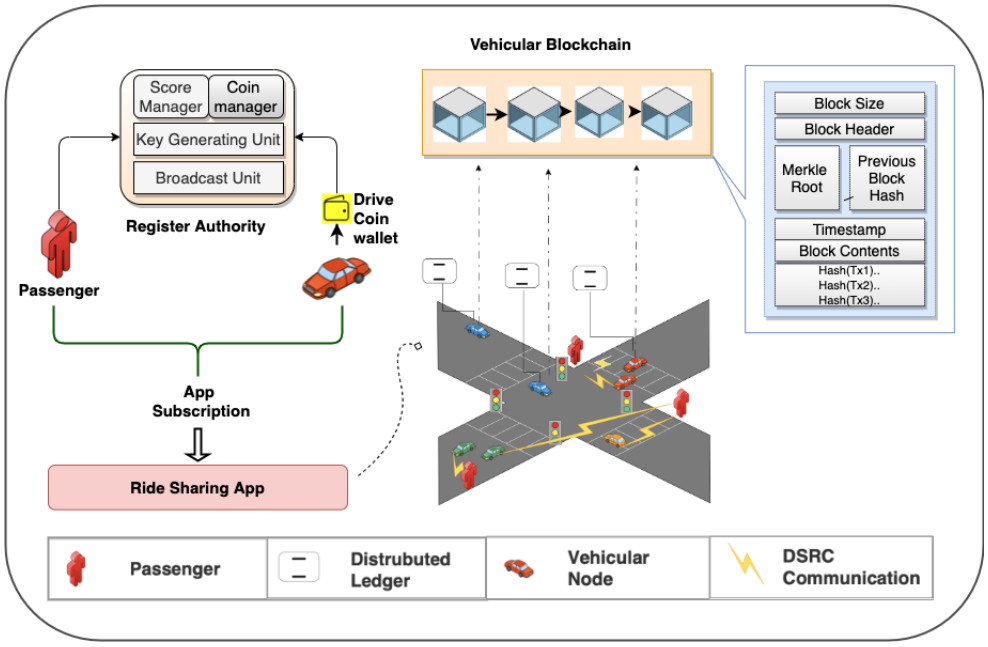


Figure 5.1: Overview of Blockchain based Vehicular Network

Register Authority (RA): The functioning of RA is as described in the section 4.3 Here we have considered few additional sub components for RA such as Key generating unit, Coin Manager, Score manager and Broadcast unit.

The **Key generating unit** is responsible for issuing public parameters and cryp-

tographic keys to the vehicles and the passengers. It keeps the database containing the linkability between the users' public keys and the account identity with a high level of security. The **Coin Manager unit** is responsible for generating new coins in the system and evaluating the coin earnings of the network and, **Score manager unit** is responsible for evaluating the S_c of each vehicle based on various parameters recorded in the network about the vehicle nodes. Finally, Broadcast unit is responsible for performing the network level broadcast in specified intervals.

Vehicular Nodes : In our blockchain environment, each vehicular node is represented as V_i where $i \in \{1, 2, 3, ..., N\}$ and N is the total number of vehicular nodes in the network. Vehicles are installed with advanced communication devices, wireless transmission modules like IEEE802.11, GPS receivers, and perform simple computations such as calculating location coordinates, determining their location by matching the electronic maps, and verifying transaction signatures.

Passenger Devices: Passengers also need the necessary hardware equipment to support communication. Any passenger device is denoted by P_j where $j \in \{1, 2, 3, 4...m\}$ and m is the total number of passenger nodes in the network. They are equipped with wireless pocket devices to call vehicles for the ride, which can be regarded as a cheap device with an electronic map and simple input/output. These devices are termed light nodes as they only participate in obtaining services and leaving service ratings. They are not part of the consensus process.

Node Categories: N represents the total number of vehicular nodes in the network, and N_{PoD} represents the number of intermediate nodes that are filtered through PoD. From the filtered nodes N_{PoD}, we arrive at a group of consensus nodes representing N_{Sc} after further filtering based on service score S_c. Finally, N_{Sc} are the entities responsible for carrying out consensus and approve transactions recorded to be added into an immutable ledger. We also represent unfiltered nodes from the

second stage as set N'_{Sc}. Table 1 summarizes all the used notations.

Permissionless Blockchain and Transaction: We consider a permissionless blockchain [37] that handles all the ride-sharing records. Any vehicle V_i or passenger P_i can use the platform to obtain services after correct registration. Transactions are the essential communication primitive in blockchain for information exchange among entities in the system model. In our model, the transaction is the record of every event such as passenger requesting for the ride, driving vehicle accepting the request with a response, passenger acknowledgment at a particular date, and time along with the amount of fee paid.

5.4 System Methodology

In this section we outline the overview of our overall system methodology.

- **System initialization**: The vehicle and passenger nodes joining the blockchain network for the first time submit their identification details such as name, address, Electronic License Plate (ELP) number for vehicles, Personal Identification Number for passengers, and other required identification details to the RA. The key generating unit of RA, in turn, assigns a pseudo-identity id_{vi} for vehicular nodes V_i and id_{pi} for passenger P_i along with generating a public-private key pair by using Elliptic-Curve Diffie–Hellman(ECDH) key agreement protocol.

  A vehicle node V_i also obtains a driving-coin wallet address WID_{vi} from the authority. The authority generates a mapping list $\{id_{vi}, PK_{vi}, SK_{vi}, WID_{vi}\}$ for each V_i and $\{id_{pi}, PK_{pi}, SK_{pi}\}$ for each passenger. This identification vector of the V_i and P_i along with its details, digitally signed by the RA are stored as a single transaction in the identification ledger.

- **Genesis Block creation**: The blockchain begins with a genesis block on top of which are stacked the successor blocks. Genesis block in our model contains empty transaction lists, a placeholder for S_c values of mining nodes. When a vehicular node joins the network for the first time, it will receive a default of 0.5, 0.0 and 0.0 for reputation r_i, error rate e_i and success rate s_i respectively. More details about the need for these parameters are provided in section 5.7.

- **Ride sharing Record**: Records of passenger REQ messages containing passenger public key, encrypted locations, and ACK messages are logged as a transaction in the distributed ledger. Records of RESP messages from vehicular nodes responding to REQ messages are also logged respectively by each responding vehicular node. These records might contain a vehicular public key, location information, REQ id to which the response is being sent, the total distance between the points of passenger and driver. These digitally signed transactions get accumulated periodically in the unconfirmed transaction pool of distributed ledger maintained by all network members.

- **Mining by Vehicular Nodes**: To avoid the mining process carried out by the massive size of a mining pool and to improve block creation time using PBFT consensus in a vehicular network, we make use of a new proof variant called PoD and a filtering technique based on service standard score value S_c for selecting miner nodes which are discussed in detail in section 5 and 6. We select an optimized number of best possible miners to compute and approve blocks in our framework during block generation.

- **Achieving consensus by adapting the existing PBFT protocol**: We use PBFT as the underlying consensus protocol. Developed by Castro and Liskov [38] in 1999, PBFT is the first BFT consensus protocol that has gained

wide recognition for practicality. In our system, after efficiently identifying the miner nodes and forming a group of best possible nodes constituting a higher S_c value sum, we use existing PBFT protocols to generate and broadcast blocks. Nodes in this stage are sequentially ordered, with one node being the primary(or the leader node with the highest S_c value) and others referred to as secondary(or the backup nodes). Every node in N_{sc} creates blocks taking turns randomly. For the very first block generation only, since all filtered nodes have the same S_c value, a leader node is chosen in this stage based on the time of joining the network.

Once a leader node successfully generates a block, it is validated by the secondary nodes, and all honest nodes help reach a consensus regarding the state of the system using the majority rule. A PBFT enabled distributed system provides a practical byzantine state machine replication that can work even when malicious nodes operate in the system, assuming that honest nodes are more than $2f + 1$ where f is the number of faulty nodes. Normal execution of the protocol can be summarized as:

1. The leader of the selected group of miners successfully generates a block and proposes it to secondary nodes through broadcast.
2. Upon receipt of a block, secondary nodes verify the block against its replica and check if the block is valid and is signed by a leader node.
3. If the block is valid, each secondary node approves the proposed block and acknowledges the leader node.
4. Upon receipt of 2/3 signatures from secondary nodes leader node commits the block and broadcasts the block to the whole network and hence adds a block permanently to the blockchain. The block contains the Merkle

root of all the transaction hashes, previous block hash, and timestamp, and current block hash. In our proposed framework, instead of voting by the whole network, only the nodes in the final consensus group N_{Sc} will send votes to the current leader. Hence reducing the communication cost of broadcasting the votes.

Incentives are released to the block creating node from the transactions validated, and faulty node if found is penalized for subverting the network by downgrading its S_c value to negative and revoking its registered keys from RA of VANET application, thereby eliminating it from taking part in the network.

- **Rating the Leader Node :** Every peer node whose transaction is included in the current block is notified when the new block is created. These nodes rate the leader node based on its efficiency in confirming the transaction. Additionally, the passengers also rate the vehicular nodes serving the rideshare platform for their quality of service. Both of these factors account for r_{vi} of each vehicular node. Consensus nodes are rated based on the outcome of block-creation. Alongside aggregated r_{vi} fetched from several nodes, success rate s_{vi} and error rate e_{vi} for each consensus node is also managed by score manager. This mechanism is discussed in more detail in section 5.7.

5.5 Proof-of-Driving (PoD) Protocol Design

PoD scheme encourages vehicular nodes to compete in a nomination process where every vehicle has a fair chance. We define the methodology step by step process as follows and provide the pseudo-code in **Algorithm 2**. Fig 5.2 depicts the high-level overview of the filtering mechanisms in the system.

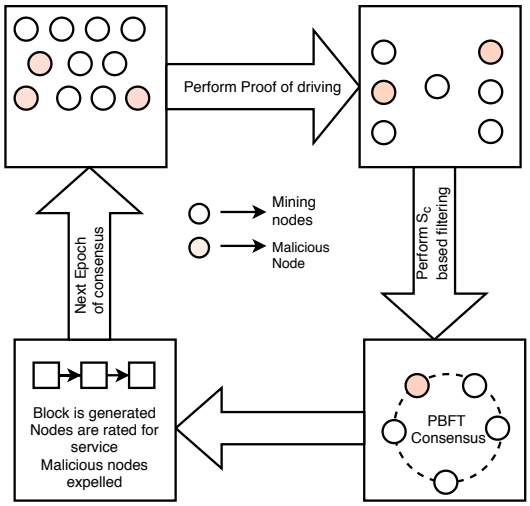


Figure 5.2: Overview of Miner Nodes Filtering Process

- **Step 1:** Actively ride hosting vehicles V_i carrying passengers on board earn driving coins ω in their driving coin wallet for every certain distance as rewards. For instance, each node earns a driving coin ω for every two miles of driving as per our design. New driving coins are created and tracked at coin generating units. This competitive process takes place just as it happens in the bitcoin network [39]. It also encourages vehicles to provide more services to passengers.

- **Step 2:** At a certain point of time, when the Broadcast unit of RA invites nominations for the miner selection process for jth round, it queries coin generating unit for average earnings of the network. Upon this request, the coin generating unit of the system calculates the average number of coins ω_{avg} that is generated so far. The target hash value for the current round is based on the average coin earnings value of the entire network. No member of the network can predict the target hash at a particular time because it would be purely based on active vehicles at that instance and the distance traveled by them.

$$\phi_{target} \rightarrow hash(\omega_{avg})$$

- **Step 3:** The Broadcast unit of RA announces the target hash ϕ_{target} to every node in the entire network. After receiving the broadcast message, node V_i runs a simple computation to check if its earned coins ω_j in the wallet pass the criteria or not. V_i calculates the value of the hash (ω_j) and checks the condition programmatically with the inbuilt application code.

$$hash(\omega_j) < \phi_{target}$$

  If the value is false, no further steps are taken, and the node continues to participate in the coming nomination processes. Else if it's true, then V_i notifies {hash (ω_j)} to the broadcast unit. A vehicular node might also choose to exempt from taking part. In all those cases, the wallet balance will be reset to 0 at the end of the current round.

- **Step 4:** After receiving verified notification messages from V_i nodes, the system chooses these nodes and then assigns them into the set of intermediate consensus node list represented by N_{PoD}.

5.6 Theorems and Proofs

Theorem 1. *Although in PoD, the reference of driving coins is taken as one of the selection criteria, the selection process is more random, and the final consensus group can never consist of a majority of faulty nodes, provided that the total number of faulty nodes in the network do not exceed f.*

Proof. Let us consider, out of the N total mining nodes f are adversaries (bad nodes)

Algorithm 2: Algorithm for Proof-of-Driving

Input:
$\omega_j \rightarrow$ coins earned by v_i at j th round where i ranges from { 1, 2, 3... n} and
$n \rightarrow$ number of active nodes for j th round
obtain the ω_{total} generated in j th round
for *timeUnit = 1 to t (time interval of j th round)* **do**
    evaluate the ω_{total} += $\omega_j \cdot n$
end
calculate the ω_{avg} for j th round = $\frac{\omega_{total}}{n}$
for the computed value ω_{avg}, compute $h(\omega_{avg})$
broadcast $targethash$ $\phi_{target} \rightarrow h(\ \omega_{avg})$
each v_i evaluates its h(ω_j)
for *i = 1 to* n **do**
    each v_i calculates $\rightarrow h(\omega_j)$
    Submit the hash to the Broadcast Unit
    if $h(\omega_i) \leq h(\omega_{avg})$ **then**
        Add v_i to N_{PoD} list
    else
        Do nothing
    end
end
Output: Broadcast Unit notifies list of nodes to be in N_{PoD} group

such that $f < \frac{N}{3}$. Let us assume we have N_h which is the number of honest nodes out of N such that $N_h = N - f$. Let N_{pod} denote the group of nodes that are filtered from PoD.

If we have to randomly select N_{pod} out of N nodes based on the proof of driving, then the probability to pick f malicious nodes is given by $P_f = \frac{f}{N}$ but $f < \frac{N}{3}$. Hence the probability of choosing bad nodes from this stage is less than $\frac{1}{3}$. Additionally probability of choosing honest nodes is $P_{N_h} = \frac{N-f}{N}$ is higher. Clearly we can state that POD protocol is efficient enough not to filter the malicious nodes into the next stage. □

Theorem 2. *With PoD, the mining in a blockchain-based vehicular network is more randomized, impartial, and dynamic than existing PoW and PoS systems in terms of resource consumption and fairness.*

Proof. In PoW, the target hash is set based on block difficulty ϕ for the current round, and the miners consume their CPU power to continuously hash the block to find a particular nonce below the target. However, in PoD, coins earned ω by hosting rideshare is proof of driving for each vehicle. In PoS, the mining is based on the user's stake or ownership of virtual currency in the blockchain system, causing partial centralization. In contrast to that in PoD, the selection of miners is randomized based on driving coin balance; therefore, the highest coin earner is not always the block validator. Let us consider N as the total active driving nodes at a given window time t. Note that N is a stochastic, dynamic, and discrete component that changes with time when VANET is quite large. Since the target hash is set as the hash of an average

number of coins ω_{avg} earned by the entire network, the coins earned at a particular time t is beyond any one's guess as one vehicle does not know about other vehicles' driving patterns and how many active vehicles are taking part in the process. PoD enables vehicles to provide proof with one-time hashing, unlike continuous hashing in PoW. Additionally, the system uses SHA256 to generate the hash of average coins, which guarantees that the attacker cannot guess the final average making it difficult for a malicious user to hijack the proof to increase the probability of being selected as a miner node. We want to emphasize that, PoD mechanism is designed by picking the best features of the existing mechanism yet preserving the security features as discussed in the future sections.

□

5.7 Service Standard Score S_c Protocol

Once the PoD process is completed, in the second stage, miner node selection takes place in the vehicular network where we make use of S_c to choose a group of final consensus nodes N_{Sc} which is a subset of N_{PoD} capable of controlling the consensus protocol.

The group size is formed by the minimum number of miners contributing to more than half of the total S_c value of the filtered mining nodes N_{PoD} from the PoD process. We try to maximize the S_c sum with the constraint on the number of nodes. From this stage, selected miners apply PBFT to decide which transactions should be involved in the proposed block and broadcast the new block to the network. The rest of the mining nodes synchronize with the newly appended block.

Algorithm 3: Algorithm for Service Score

Input: $r_{vi} \rightarrow$ reputation of v_i, $e_{vi} \rightarrow$ historic error rate of v_i
$s_{vi} \rightarrow$ historic success rate of v_i
$n_{pod} \rightarrow$ number of nodes filtered through POD for j th round
calculate the S_c corresponding V_i node for j th round
for `i = 1 to` n_{pod} **do**
    calculate the $Sc_i = r_{vi}$-e_{vi}+s_{vi}
    broadcast Sc_i
end
sort list of S_c value for n_{PoD}
calculate target S_c= 1+(0.50* Total S_c)
currentSum = S_c[n-1]
loop through remaining S_c values
for $i = n - 2$ `to 0` **do**
    if *currentSum* $\leq$ *Target* S_c **then**
        set currentSum = currentSum+S_c[i]
        Add node i to the List N_{Sc}
    else
        skip node
    end
end
Output: Broadcast list of node indexes selected as N_{Sc} group

5.7.1 S_c Aggregation

A node's trustworthiness is defined by S_c it holds in return for generating a successful block in the past. Such a score reflects the degree of trust that other peers in the community have based on their past experiences, and it determines the node's ability to obtain leadership authority. We identify three important factors for S_c evaluation as (1) Reputation r_{vi} (2) Error rate e_{vi} and (3) Success rate s_{vi}. We illustrate the importance of these parameters and explain how they can be calculated in our design.

- **Reputation** r_{vi}: When a consensus process is completed, and the block is published in the blockchain successfully, every consensus node is rated by the peer nodes either positively or negatively for each transaction validated based on the honest behavior and efficiency that the node has demonstrated in creating a valid block. However, it is also possible that malicious peers can attack the rating process by deliberately giving high or low scores for the consensus nodes. Consequently, the consensus nodes' reputation as miners is either increased or decreased. Hence these ratings are aggregated by the score manager unit of RA to obtain the weighted average rating over multiple transactions, which is considered ground truth for a particular node. Lastly, due to the limited number of malicious nodes, as discussed in section 5.8, these unfair ratings can hardly disrupt the system. The calculation of aggregation can be represented, as shown below:

$$r_{vi} = \frac{1}{t_x} \sum_{i=1}^{t_x} m \text{ - } \frac{1}{t_x} \sum_{i=1}^{t_x} n \in [0,1]$$

where r_{vi} is the reputation value of vehicle V_i based on n^{th} consensus round and m and n are the number of positive and negative ratings, t_x is the number of transactions validated by the consensus nodes. r_{vi} is normalized to a value

between [0,1]. When a node first joins the network, it gets a default reputation of 0.5.

- **Error rate** e_{vi}: We define the error rate factor e_i for node V_i as the ratio of the number of times a node has failed to generate a valid block e to the total number of attempts made by this node A_{total} to generate blocks. We define the error rate of the nodes which has generated no blocks as 0. This parameter has a negative impact on the S_c of a node

$$e_{vi} = \frac{e}{A_{total}} \in [0, 1]$$

- **Success rate** s_i : We define the success rate factor s_i for node V_i as the ratio of the number of times a node has created block successfully S to the total number of blocks B_{total}. We define the success rate of the nodes which has generated no blocks as 0.

$$s_{vi} = \frac{S}{B_{total}} \in [0, 1]$$

Given the blockchain, the S_c score of any miner can be calculated at any point in time. Accordingly, each miner maintains its own copy of the S_c score of all the miners. Algorithm 3 shows how the S_c is calculated for a miner.

5.7.2 Sc Score based Grouping

Now that we have S_c calculated for each of the miner nodes, our goal is to select a group of nodes N_{Sc} from the set of filtered nodes N_{PoD} to maximize S_c value of the group, subject to the restriction on the number of nodes. In other words, given a list of S_c values for each node V_i, we need to find the maximum possible sum of S_c such that the sum of S_c of selected nodes N_{Sc} will be greater than combining S_c values of

not-selected nodes N'_{Sc} from this stage. We provide the pseudo-code for implementing the above-described methodology in algorithm 3.

In the first step, the target S_c value for the group to be selected is set by considering above 50% of the total S_c value. The group of nodes to be selected N_{Sc} would have higher S_c sum compared to the combined S_c sum value of unfiltered nodes N'_{Sc}.

In the second step, we are sorting the nodes based on S_c score first, and then the score manager unit identifies the set of nodes programmatically looping through the S_c values of the nodes, and by checking the cumulative S_c sum value until target S_c sum set before is reached. However, for the very first round, as the S_c is the same for all the nodes, the second filter criteria based on the timestamp of joining the network is considered. We analyze that complexity of the S_c filtering scheme is $O(NlogN)$. Finally, at the end of this stage, N_{Sc} a group of nodes performing PBFT is obtained and notified to the broadcast unit, which disseminates the information to the rest of the network.

5.8 Defense against attacks on PoD and Sc mechanism

Various attacks can be launched on the proposed model. Hence, this section focuses on explaining how the proposed methods can defend themselves from those known attacks. Much like the PoW consensus, the PoD consensus mechanism as well implements a target hash-based mining mechanism. However, unlike in PoW, which gives the ability for the attacker to break the system by owning higher computation power, PoD does not give way for it. Instead, an attacker has to contribute to the network by hosting ride shares and gaining a driving balance in their wallet. We also would

like to recall here that the Service score S_c of vehicular miner nodes with honest behavior in the network builds essentially on its continued participation and regular contribution to block creation in the entire blockchain system. Below we describe the details of the effect and defense of various attacks.

5.8.1 Security against Internal Attacks

Effect: Internal node is an authenticated member of the network, which, if turned malicious, severely impacts the safety in the proposed system. There could be an individual or multiple colluding malicious nodes working together to break into the system. In PoW based systems, if an internal malicious node owns the temporary majority of the entire network's computing power, it breaks the PoW based distributed systems.

Defense: In the proposed system model, we rely on computing power and have incorporated randomization through proof of driving protocol. Consequently, there is no time guarantee for an internal adversary when it gets selected as an intermediate miner, as discussed in theorem 1. Due to randomization, it introduces a dilemma for an adversary whether to earn more coins or earn more service scores. Either of these would be unpredictable because the first stage of filtering entirely depends on the average coin earnings of the network, which one cannot guess accurately at a given point in time. This also depends on the network as to how many nodes are active. Therefore, even if an insider adversary exists, his/her chance of becoming the mining node is no larger than other users.

5.8.2 Security against External Attack

Effect: Here, we consider a specific case of an external attack in one of the existing systems based on the PoS mechanism. If the number of coins owned by a single miner is more than 50% of the entire blockchain network, an external adversary can directly compromise such a node. As subsequent rounds of consensus are solely dependant on the highest staked node in the network, controlling such nodes can cause arbitrarily manipulating and modifying the blockchain information.

Defense: In the proposed model, it is not the case due to randomization in the filtering process in the first stage. We show that the selection of nodes is diverse across various coin balances in Fig 5.3(b). The highest coins earned node is not always getting filtered, eliminating partial centralization and the described attack scenario. Additionally, it is impractical for one vehicle node to earn more than 50% driving coins than the rest at a given window.

5.8.3 Security against Attacking the Leader Node

Effect: The proposed model adopts PBFT for consensus in which block creation is done by leader node once the miner group N_{sc} is finalized. An external observer may attack the potential leader node in order to control the block creation process. This attack is similar to the attack on the highest staked node.

Defense: This attack is ineffective for the proposed model due to the active change of the leader nodes. After fixed rounds of consensus, the leader node will be changed. The leader for the subsequent consensus rounds is randomly selected based on produced proof of driving and service score protocols. Since the selection of the new leader depends only on this, no node knows the next leader in advance. Thus, the

probability of a successful attack is significantly reduced.

5.8.4 PoD and Sc Attack Resilience

Here, we consider a passive approach where a miner node that intends to break the system in the future may behave honestly by contributing to the blockchain system creating good blocks for some period, which allows it to have a higher reputation. In the worst-case scenario, one of such passive attacker nodes might successfully get filtered through the PoD mechanism and gain its spot in the final consensus group N_{sc} to create blocks in the system. If N_{PoD} is the number of nodes chosen from the first stage of filtering and Sc_i is their corresponding service score, our model ensures safety against such attacker node being a leader if (1) $\frac{2}{3}$rd nodes from the consensus group N_{Sc} are honest (2) the sum of service standard score of few attacker nodes Sc_a is less than the sum of service standard score of honest nodes. i.e., $Sc_a < \frac{1}{3} \sum_{i=1}^{N_{PoD}} Sc_i$. This clause holds good because the honest mining node's service score would be much higher due to the complimenting weights for success s_{vi} and error rates e_{vi} in service score S_c calculation as per algorithm 3. In summary, although the attacker breaks into the system, the system can be lively and resilient if both the conditions mentioned above are true. Otherwise, block creation will be hampered.

5.9 Experimental Analysis

We use Java SE 8 on a hardware platform with Windows 7 OS to conduct the experiments with the following specification: Intel(R) Core(TM) i7-9700 CPU@3.00 GHz, 8.00GB RAM. To study the effectiveness of our proposed protocol, we mainly focus on

1. How effectively PoD and S_c strategies can reduce the number of vehicle mining

nodes from the pool of miners compared to traditional PBFT? Are the highest coins earner always get filtered?

2. What is the S_c value of group of selected nodes N_{Sc} versus not-selected nodes N'_{Sc}. Is there a considerable number of miner optimization?

3. Is the proposed mechanism PoD secure against excessively driving nodes?

4. Is the PoD and S_c mechanism secure against the infiltration of malicious nodes?

5. How scalable the proposed system is in terms of PoD and S_c protocols?

The rest of the experimental section is organized by answering the above question with detailed analysis and discussion. To conduct our experiments, we collected data[1] from one of the traffic surveys that has recorded the distribution of vehicles across various periods of the day in the interval of one hour with varying speeds across four different regions. Using this data reference, we simulate the vehicular mining node distribution for all four regions with the same speed pattern from 10:00 to 17:00 during a particular day of the month as captured by the survey. However, the time interval for the mining process is a configurable parameter tailored to the application's needs.

We recall here that in our model, for every two miles of distance traveled by mining node, it earns one coin. Hence this data set is very suitable for our experiments as we can accurately derive the number of coins earned based on the speed per hour input.

5.9.1 Analyzing PoD based Filtering Process

To validate the feasibility of the proposed PoD selection strategy, we perform this experiment in several rounds during the different periods of the day. We observe that

[1]https://data.world/cityofaustin/et93-wr2y

from Fig 5.3(a), the number of nodes competing for mining varies each time, reaching the peak at 13:00 with over 688 nodes as captured by the traffic survey data.

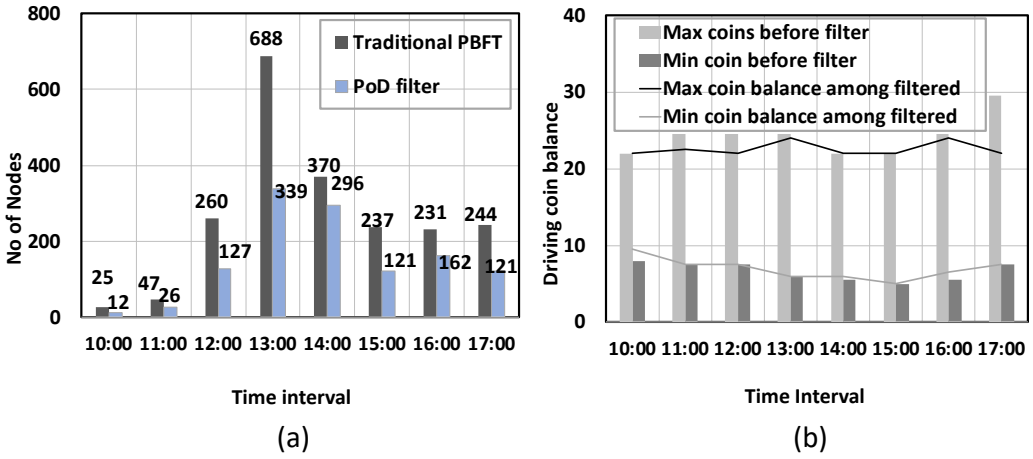


Figure 5.3: Experimental Results for (a) Number of miner selected by Proof of Driving algorithm during different time intervals (b) Distribution of Coin Balance Before and After the Filter Process

We first simulate speed data referring to the data set for larger values of N, where N is the number of potential miner nodes from across different regions during the different time intervals of the day. Then this data is fed into the PoD to compute the coin earnings of each vehicle node. Here we recall that coins are earned based on distance traveled. Finally, once the computation is complete, we display the number of filtered nodes.

At this first stage of filtering through PoD, we observe how our proposed algorithm only selects a subset of mining nodes based on the hash of coins earned for the next stage of filtering, and we represent the results in Fig 5.3(a).

It can be inferred that there is no significant difference in the filtering process when the mining pool is smaller during the early hours of the day. However, as the mining pool grows at around 13:00, which is as in the real world scenario, we see that the significant number of miner nodes are filtered and almost halved by implementing this strategy, which is an excellent pre-requisite to run PBFT with a fewer number

of nodes.

We also calculate the highest and lowest coin earnings at each interval before and after filtering. Fig 5.3(b) represents the minimum and maximum driving coin balance of different groups of nodes before and after the filtering process. From this visualization, we can infer that, unlike PoS, which only selects the highest staked nodes(here stakes are referred to as driving coin balance), in our selection process, there is randomness. The PoD does not always filter the highest coins earner but filters a range of nodes with different coin balances.

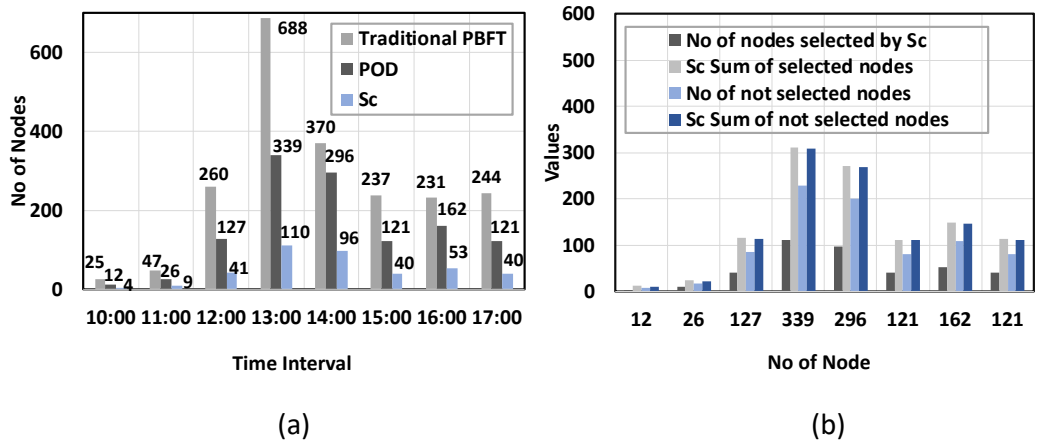


Figure 5.4: (a) Comparison of the Miner nodes selected by using PBFT alone versus using Proof of Driving and S_c filter (b) Visualization of number of selected nodes N_{Sc} vs not-selected nodes N'_{Sc}, after running S_c protocol

5.9.2 Analyzing S_c based Filtering

In this experiment, to validate the effectiveness of the S_c algorithm, we take the list of filtered nodes from PoD with simulated S_c values ranging from 0 to 2. We trace how these nodes are getting further filtered based on our proposed algorithm 3. This experiment is also performed in several rounds during different periods of the day. The results are represented graphically in Fig 5.4(a) by comparing the number of

miners considered by traditional PBFT and our proposed filtering strategies. From Fig 5.4(a), we can infer that the S_c based filtering strategy provides a better reduction in the number of miners, based on group S_c value, which helps PBFT consensus to execute in a short period.

In order to visualize the optimum number nodes and their corresponding sum of S_c, we plot a bar graph, as shown in Fig 5.4(b). Here we can see that the protocol is selecting such groups containing an optimized number of nodes N_{Sc} that makes up the total S_c value more than the group of not-selected nodes N'_{Sc}. Also, it is worth noticing that the number of selected nodes N_{Sc} is always much lesser than the number of not-selected nodes N'_{Sc} as we maximize the total S_c values with a minimum number of nodes to be selected.

5.9.3 Analyzing the Security of POD and Sc Against Malicious Activities

In this subsection, we present an analysis of the security of our proposed method. Since the main criteria to be selected as miner node is to drive the vehicle honestly, to test its security, first, we experiment to see if any driver can win the selection process by intentionally driving more than the system average. Then we conduct another analysis against infiltration attacks of malicious nodes.

In section 5.8, we have discussed that PoD combined with Sc is safe and live as long as the combined service score of attackers is below the service score of honest nodes; In this subsection, we experimentally prove the analysis by visualizing the outcomes in different consensus rounds. We have executed this experiment in four rounds with a varying number of total miner nodes taking part in the PoD mechanism. We have also simulated the presence of $\frac{1}{3}$ of malicious nodes in every round that is passive

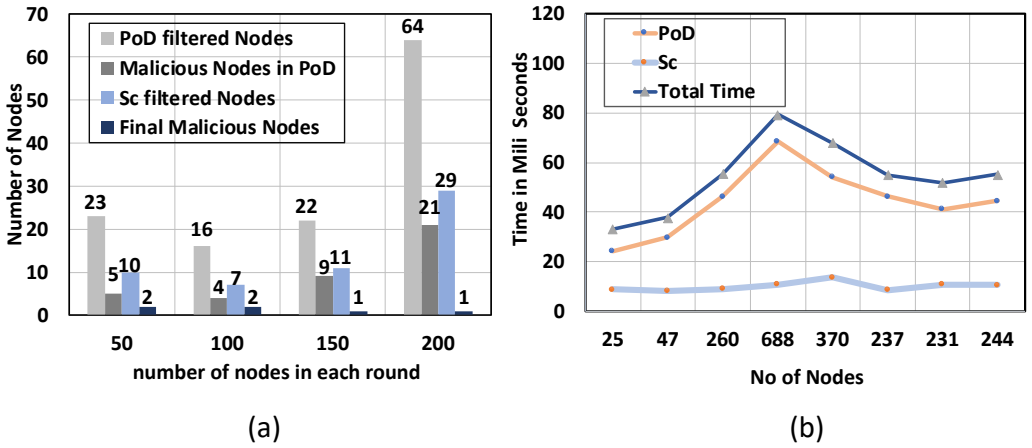


Figure 5.5: (a) Analysis of PoD and Sc mechanism's efficiency in filtering out malicious nodes (b) Latency for one round of PoD and S_c with a varying number of vehicular miners during different time intervals

and are actively taking part in ride-sharing service to gain a good reputation with the sole intentions of infiltrating the final consensus group. Figure 5.5(a) represents the results, which shows that, although there are few of the malicious nodes that are getting filtered by PoD, the application of the Sc filtering mechanism in the second stage is ensuring that the final consensus group N_{sc} is majorly containing nodes of higher service scores. Results also show a minute number of malicious nodes in the final consensus group represented by the dark blue bar in Fig 5.5(a). However, it is worth noting that, at this stage, the application of PBFT would ensure the system is fault-tolerant and can perform efficiently.

5.9.4 Analyzing the Time Consumption to Run PoD and S_c Filtering Process

We also measure the latency of our proposed schemes to select a certain number of nodes from a pool of nodes ranging at various periods of the day. In Fig 5.5(b), the curve in the middle represents the trend w.r.t time consumed by PoD, and the

bottom-most curve represents the S_c scheme to select nodes from a large pool of miner nodes. The topmost curve is the total time spent to run both the algorithms during one round of miner selection. The results show the relationship between the running time and the network scales, i.e., the number of miners, which is close to linear as the network grows. It is evident that the time consumption is minimal; therefore, the protocol can be scaled to many nodes.

Although there is overhead time for using PoD and S_c filtering before PBFT consensus, our protocol reduced the mining group size by approximately 84 percent (without compromising on the quality of filtered nodes during peak time as shown in fig 5.4(a). We have set the time interval for the miner selection process to take place every one hour. During this time, a group of selected vehicular nodes will be taking turns in creating subsequent blocks. Therefore, the overhead time associated with the PoD and S_c filtering technique is applicable only once at the beginning of every interval. This total overhead time is shown in Fig 5.5(b) is minimal (approx. 33ms & 79ms for 25 and 700 nodes, respectively) compared to the total consensus time required in traditional PBFT without any filtering technique.

5.10 Summary

In this chapter, we proposed an efficient and effective miner node selection strategy for a VANET application based on blockchain. More specifically, we proposed the Proof of Driving protocol introducing the driving coins associating it with one of the vehicle features such as distance traveled to achieve more randomness in selecting the miner nodes. Additionally, we also proposed the service score-based protocol to ensure that the selected nodes are of high reputation, low error rate, and high success rate w.r.t successful block mining and ensured that the quality of the mining node

is not compromised while aiming to achieve randomness. In the next chapter, we will look into internal attacks such as the Blackhole attack and study how promising features of blockchain such as decentralization, consistency, and tamper-proofing can be leveraged to help cope with the trust management problems in vehicular networks.

Chapter 6

Blockchain based Trust Management for VANET Routing Protocol

In the earlier chapters, we mainly focused on exploring the blockchain platform for VANET applications to store ride-sharing data using smart contracts and introduced novel filtering techniques for consensus algorithms. In this chapter, we focus primarily on the insider threats such as Blackhole attacks carried out in V2V communication while disseminating critical information. This can disrupt the networks' average performance and prevent transmission between vehicles entirely. To address this problem, we present a trust score management system based on blockchain and evaluate the efficiency of the system through various experiments.

6.1 Background on Routing Protocol

In a V2X communication-based application, vehicles highly rely on routing protocols that determine the path for packet transmission from source to destination. Regard-

less of the application considered, the underlying routing mechanism at the network layer is crucial and significantly affects the applications' overall performance. Out of the many routing algorithms, Ad-hoc On-Demand Vector (AODV) [40] is an extensively adopted reactive routing protocol in a dynamically changing network such as VANET [35,41]. Additionally, less memory consumption for processing makes AODV the best fit for resource-limited vehicular nodes.

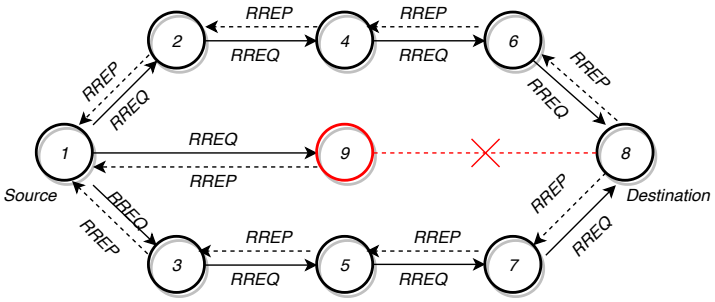


Figure 6.1: Balckhole Attack in AODV Protocol

A route discovery process is initiated in AODV whenever a node wants to communicate with other nodes. AODV protocol uses three control messages such as Route Request (RREQ), Route Reply (RREP), and Route Error (RERR), as shown in Fig. 6.1. RREQ packets are broadcast to the nodes in the network by the source to find a path. All the other nodes that receive the RREQ packet keep transmitting them until they find a fresh enough route to the destination. On receiving RREQ, if the node is the destination or if the intermediate node has a new route to the destination, it sends the RREP packet back to the source. The Hop count of every node increases by one on receipt of the RREQ message, and route information entry is updated with new data by intermediate nodes on receipt of RREP messages. However, if the link is broken between two nodes, the RRER message is sent back to the source node via the reverse path. Each node on receiving RRER invalidates the route in their table to an

unreachable destination. A node increases its destination sequence number each time a new RREQ, RREP messages are sent. Destination Sequence Number (DSN) is a 32-bit integer associated with every route and is used to decide a particular route's freshness. The higher the sequence number, the fresher the route is.

Although AODV has been around for quite some time, few security issues make it vulnerable to various attacks. An internal malicious node uses the AODV routing protocol's vulnerability for advertising itself for having the shortest path to the destination node irrespective of its routing table entry. It intercepts the passing through packets and drops all the packets transmitted from the source node, causing a disconnect in the network. This attack is known as a **blackhole attack** [8]. In Fig. 6.1, node 9 represents a member launching a blackhole attack. This attack is perceived as a denial-of-service attack by a node or a router. It either refuses to participate in the network or drops the information packets instead of transmitting them. A blackhole is a hazardous attack as the data packets containing important information are lost permanently during its transmission to the destination. The situation worsens when multiple blackhole nodes exist in the network. Consequently, the network may become unavailable and may lead to crashes and congestion in road traffic information-based VANET applications.

6.2 Motivation

Trust management models are usually adopted to achieve a more efficient packet forwarding process and mitigate packet drop attacks in VANETs. Trust-based solutions help to detect selfish nodes that act as blackholes in the network. Authors of [42] propose computing a distrust level for every neighbor performing as a blackhole through a watchdog technique. A computed distrust level will be sent to the cluster head and,

in turn, delivered to a trusted third party, which revokes the attacker's certificate. Nevertheless, untrustworthy intermediate vehicles can modify the message containing trust values. They can even generate and insert a new trust value in the VANET, causing broadcast tampering attacks [43]. Furthermore, a certificate revoking the third party may also get compromised, subverting the trust model. Hence, a trust model designed to mitigate blackhole attacks should also be able to resist various attacks and preserve vehicles' privacy simultaneously.

In this context, several initiatives have been launched recently to investigate the suitability of trustless and decentralized ledger technology (DLT), also known as the blockchain, in securing vehicles' trust scores.

6.3 Contribution

In this work, we have designed a consortium blockchain consisting of authorized nodes (RSU) and vehicular nodes. In this decentralized system, the trustworthiness of an individual node is majorly based on quality metrics concerning routing protocol such as packet delivery ratio (PDR), the time difference in response to new route query RREQ and difference in destination sequence number (DSN). The existing AODV routing protocol has been modified to calculate the trust scores of neighbor nodes, and it is referred to as TAODV in this chapter. The vehicular nodes, which are part of the blockchain, monitor the neighbor nodes to evaluate the trust scores. Calculated trust scores are logged as a transaction in the distributed ledger. The authorized validators perform the trust score aggregation from these transactions. Apart from that, they also update the blacklist node table based on the pre-configured trust-score threshold. The blacklisted node table and aggregated trust score for each individual node are immutable by malicious nodes, transparent, and quickly distributed to all

the nodes via the blockchain.

The main contribution of this paper is summarized as the following;

- First, we exploit blockchain features to implement a blockchain-based two-level trust score system as a solution to detect and blacklist multiple blackhole nodes from the network much different from conventional methods of elimination.
- Second, we design the processing logic for decentralized transaction pool and trust score aggregation in a VANET system.
- Finally, we present the results in terms of the impact of the proposed blockchain-based trust model on network metrics such as throughput rate and packet drop ratio through simulation.

6.4 Proposed System Model

This section gives an overview of the proposed trust score management system based on blockchain as described in our published manuscript [44]. Then we introduce the main components of the system architecture, as illustrated in Fig.6.2, as well as the threat model for the system. Finally, we present the system methodology in detail.

6.4.1 Components of System Model

Consortium Blockchain Network: The consortium blockchain network [45] is the core of our proposed scheme. In a consortium blockchain, the nodes that participate in the consensus are pre-authorized, and they determine the generation of each block. In this design, RSU is a pre-authorized node. RSU is granted the right to write data into the blockchain and participate in the consensus. These are considered as full nodes which authoritatively verify all transactions in the network [4]. On the other

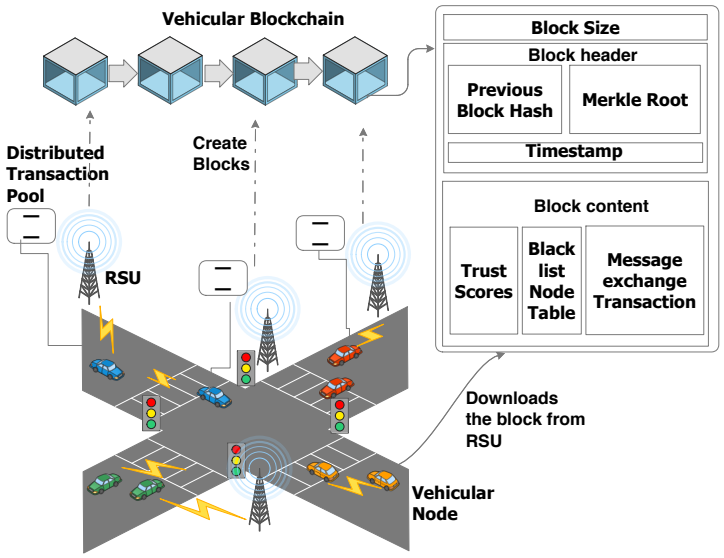


Figure 6.2: Consortium Blockchain-based VANET System Architecture

hand, a vehicle is a lightweight node that can access the data replicated on the RSU storage, but it does not participate in the consensus.

Local storage in the RSU is responsible for collecting data uploaded by the vehicular nodes and obtaining data shared by other RSU. The consensus mechanism resolves the problem of mutual trust between the nodes in the system. Furthermore, the trust score information in one country need not be shared with other countries if border-crossing traffic is not allowed between those countries. Hence for simplicity, we consider regional blockchain [46] specific to a geographic area maintained by the roadside units (RSU). Incorporating regional blockchain into the design for VANETs ensures that the blockchain is shared among nodes in a geographically bounded area.

Blocks: A block of a blockchain consists of a header and a body. The previous block's hash, timestamp, and Merkle root of transactions are included in the block's header. The block body consists of a list of trust score messages that behave as transactions uploaded by vehicular nodes. Apart from that, the body also holds the aggregated trust scores and a blacklisted node table.

Transactions T_x: In a blockchain-based VANET, records of message exchanged, services utilized, etc., can be a part of each transaction. In our model, transactions are specifically referred to as uploading trust scores of neighbor vehicular nodes to the nearest RSU, about 1000m in the communication range.

Road Side Units (RSU): We consider RSU as the edge nodes upgraded to have computational capabilities and storage space. In our model, RSU maintains local storage that collects transactions. After verifying the transaction signature, transactions are broadcast to other RSU. RSU also acts as an aggregator for calculating cumulative trust values for a vehicular node and performs block creation to append new blocks to the blockchain.

Vehicular Nodes: These are vehicles equipped with sensors and OBU (which is in charge of all communication and computation tasks) that can communicate with each other and RSU through radio. In our blockchain-based VANET environment, each vehicular node is represented as V_i where $i \in 1, 2, 3, 4..N$ and N is the total number of vehicular nodes in the network. These nodes are assumed to be lightweight and are not part of the block creation process. Information exchange between any two nodes in VANET occurs through dedicated short-range communication (DSRC) [43] radio protocol via which vehicles exchange messages with nearby vehicles in V2V V2I connectivity mode. Vehicular nodes are responsible for uploading transactions into the shared ledger maintained by RSUs. Vehicular nodes have the lowest security level.

6.4.2 Threat Model

Both RSUs and vehicular nodes are vulnerable to attacks, which can cause network performance deterioration. RSUs are considered semi-trusted with a medium level of security. Some of the vehicular nodes' operations may as well be taken control of by the adversaries. These malicious nodes may act individually or in collaboration

to drop the packets passing through them. Although all the communication between the RSUs is assumed to occur via a secured channel, we consider the following types of attacks can be launched to jeopardize the running system.

- Defaming or Bad-mouthing attack: It is possible to sense a bad-mouthing attack in this model, which means that the vehicle can generate a false trust score for an honest vehicle and upload the transaction to the distributed ledger.
- Identity Spoofing attack: Vehicular nodes may try to spoof the identity of the other nodes in the network and try to upload the trust scores to the blockchain.
- Tampering Blacklist Node table: Malicious internal nodes might try to add/delete or modify the blacklisted node table to hamper the system's integrity.
- Byzantines RSU's: Some of the RSUs may act maliciously or might be under the control of external attackers during the validation process to cause damage to the network.

While we consider the attackers not to control all the nodes within the network causing eclipse attack, [47], we build our model with an assumption of having about 25% of the malicious nodes in the network and design countermeasures against the blackhole attack.

6.4.3 Design Goals

Under the threat model defined, our goal is to design tamper-proof trust scores and blacklist node tables in a vehicular network, which is an effective and efficient trust score management system with the following key requirements.

- The proposed system should be (1) scalable to support a very high range of vehicular nodes that join the network (2) transparent so that all the autho-

rized members of the system should have access to the same immutable records (3) tamper-resistant so that it ensures the integrity of stored trust scores and blacklist node tables. (4) enabled to audit to produce tamper-proof evidence.

- The proposed system should be free from a single point of failure (SPOC). Thus it necessitates the need for incorporating decentralization in our design so that no single entity is holding control of the entire system.

- The processing and execution speed of the proposed system should be in the order of a few milliseconds so that each transaction is processed and an updated trust score is available to the entire system without having to wait too much. Fast dissemination of trust scores should minimize the routing overhead in VANETs.

- The cost of data storage associated with the proposed system should be of an acceptable range which is a crucial design requirement.

6.4.4 Overview

In a VANET, every vehicular node maintains a routing table for known destinations. The route is updated for the unknown destinations using RREQ and RREP messages over the Trusted AODV routing protocol. In this protocol, the node can promiscuously monitor its neighbor node for generating trust. A promiscuous mode is where an honest node taps the packets being forwarded by its neighboring node so that a node can determine whether an adjacent node forwards a packet or drops [48]. A caching mechanism is implemented in the TAODV protocol to verify that a neighboring node forwards packets. To determine if it is the same packet, the node verifies the tapped packet with the cached packets. If cached packets cannot be tapped from

their neighbor, they are considered to be dropped. We make use of this protocol in our design.

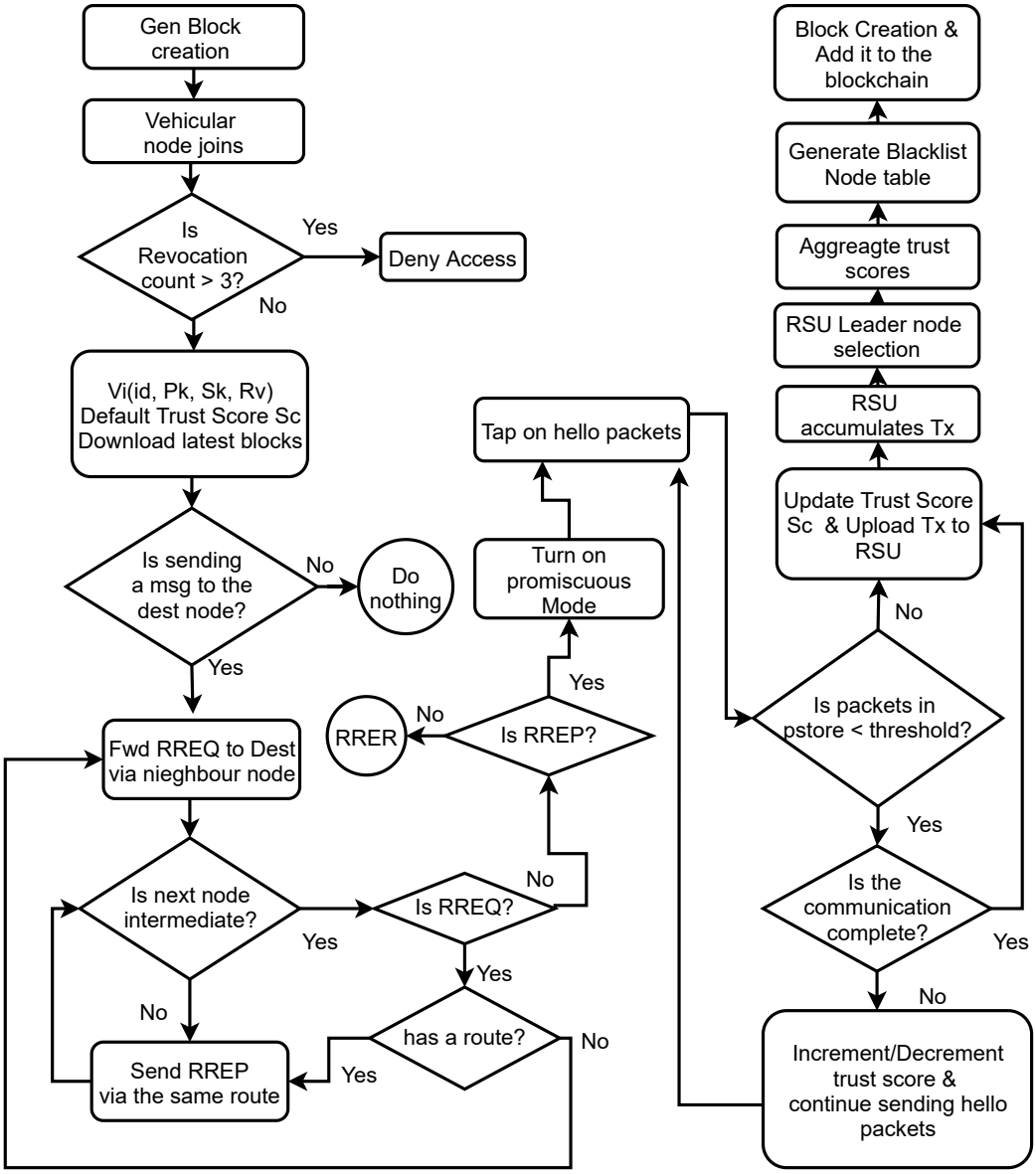


Figure 6.3: Flowchart of the proposed blockchain based VANET trust score system

In this method, the node transmits dummy packets to its neighbor node over the UDP transport layer protocol. Using promiscuous mode, the node can judge its neighbor node for a pre-determined short duration and assign a trust score. In our experiments, we have set the time limit to 120s.

Instead of broadcasting trust score by each node to the network and causing computation overhead of trust value aggregation, in our proposed model, the trust score is managed in a decentralized manner. It is uploaded as a transaction to the nearest RSU. RSU verifies the signature to validate that the message is from an authenticated node and further broadcasts the transactions to other RSUs, thus maintaining a distributed ledger.

Since the distributed ledger system's state has to be agreed by the peers, and the consensus has to be achieved. RSUs in our model, which are pre-selected validator nodes, follow the Practical Byzantine Fault Tolerant (PBFT) consensus mechanism to reach consensus. PBFT is an improved version of Byzantine Fault Tolerance that ensures an agreement regardless of malicious behaviors on the part of some participating nodes [38]. After every interval of window time t_i, a leader node is chosen that aggregates all the transactions together from the processed transaction pool to publish a new block of data. This block contains the previous block header, current block hash, Merkle root [49] of transaction records, aggregated trust scores of nodes, and a blacklisted node table.

As we are dealing with essential event messages, reliable and quick message dissemination is of high priority. Vehicular nodes, which are light in the network, periodically download the latest trust scores and blacklisted node table from the blockchain to refer to before performing any message dissemination through the network nodes. Thus insider attack launching nodes are detected and eliminated based on trust score. A safe, reliable, and tamper-free route to communicate messages in the network is ensured via blockchain technology.

6.4.5 System Methodology

In this section, we outline the overall system methodology of our proposed framework step by step.

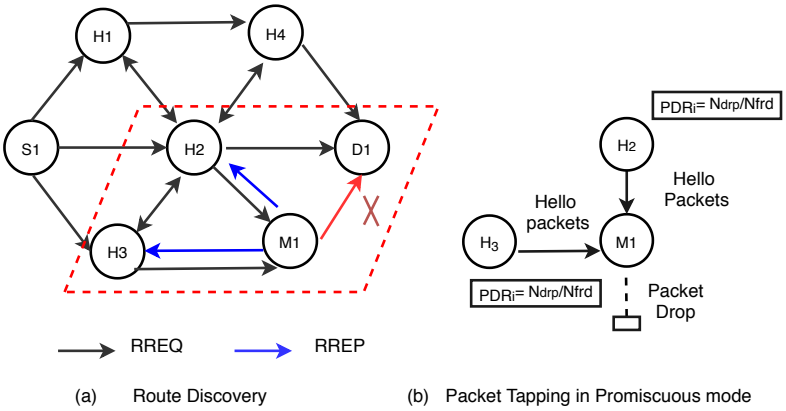


Figure 6.4: Flow of Route Discovery and Packet Tapping in Promiscuous Mode

1. **System Initialization**: For the first time, the vehicular nodes joining the blockchain network submit their identification details such as name, address, Electronic License Plate (ELP) number of vehicles, and other required identification details to the RSU. It, in turn, assigns a pseudo-identity id_{vi}, which is a unique number for each vehicular node V_i along with generating a public-private key pair by using Elliptic-Curve Diffie-Hellman(ECDH) key agreement protocol. Each license plate is also mapped to a renewal count RC_{vi} that counts the number of times a node has re-registered in the network. The RSU generates a mapping list $\{id_{vi}, PK_{vi}, SK_{vi}, RC_{vi}\}$ for each V_i. This identification vector of the V_i is generated whenever a vehicular node rejoins the network incrementing RC_{vi}. It will be digitally signed by the RSU and stored as a single transaction in the identification ledger.

2. **Trust Score Assignment**: As an additional initialization step, at the time of joining the network, each vehicular node V_i gets a default trust score. For

simplicity, we have considered the default trust score as 0.5 on a scale of 1.0. The value of the default trust score is solely a design decision at the time of system initialization. Assigning default trust scores to each node is an essential factor that draws a boundary between honest and dishonest peers and new network members. Trust scores vary dynamically with parameters such as time and packet delivery ratio. As the packet delivery ratio diminishes below a stipulated threshold, the trust score is dropped to 0.0. Consequently, the corresponding vehicular node is blacklisted. Thus trust scores of 0.5 signify a newly joined member of the network. Trust scores beyond 0.5 indicate more significant levels of trustworthiness of the member node.

3. **Genesis Block Creation**: The blockchain begins with a genesis block on top of which the successive blocks are stacked. Genesis block contains a previous block hash as "0", blacklisted node table, and transaction lists as null along with the current timestamp placeholder, the placeholder for the highest destination sequence number. When V_i joins the network, then it only knows the genesis block. V_i will have to the latest block from the nearest RSU containing blacklisted node table, aggregated trust scores as per the latest communication.

4. **Sending Route Discovery**: Let us assume the node S_1 to be the source node desiring to communicate with node D_1 as in Fig 6.4(a). Thus, as per AODV protocol, node S_1 floods an RREQ packet in the network and waits for the RREP packet to obtain a fresh route to the destination node D_1.

   All nodes forward the request further in the network until a new route notification is returned. When the RREQ reached M_1, it returns RREP with the highest destination sequence number (DSN) to its neighbor nodes H3 and H2,

as shown in Fig. 6.4(a). When the RREP packet is sent from an intermediate node M_1, nodes preceding the node which sent the RREP packet, i.e., H_3 and H_2 in Fig. 6.4(b) gets alerted to certify the RREP sending node. Node H_3 performs the preliminary checks from the RREP message to check two conditions. (1) DSN received vs. current maximum DSN recorded in the blockchain network. (2) Timestamp difference between RREP and RREQ.

5. **Entering Promiscuous Mode**: Once the preconditions are met, node H_3 switches on its promiscuous mode to tap onto M_1. Promiscuous mode [48] is how a node can overhear the packets transmitted by its neighbor node by tapping it as represented by pseudo-code algorithm 3 & 4. Node H_3 sends a series of hello packets to the destination node D_1 via node M_1.

   Variable *pstore* holds the list of all hello packets sent out from node H_3. Information can be fetched by calling packetLookUp method on each packet. Algorithm 3 loops through each packet in the pstore and calls MethodTap on it. Struct *hdr* holds the header information of each packet sent out from node M_1. $MethodTap$ verifies if the packet sent out from M_1 matches the packet stored in source node's pstore to certify that the node is not malicious. If it matches, it deletes the packet from pstore invoking deletePacket on that particular packet. We make use of these algorithms for VANETs, which allows honest nodes H_3 and H_2 to intercept and read neighbor node M_1 within its range, sending network packets.

Algorithm 4: Caller Method

Input: *tstore* → an object of *trustStore* of node *i* contains methods *trustUpdate* and *trustLookUp*
pstore → an object of *packetStore* that contains a list of packets stored in cache, has access to *packetLookUp* method and has access to *deletePacket* method
*Packet *p* → an object of packet that has uid, src, dest, fwrdId
initialize default *trust* of node i → 0.5
loop through every packet in *pstore* and invoke MethodTap
for `Packet *p in` *pstore* **do**
    invoke MethodTap
end
Output: *tstore* → trust score obj of neighbouring node i

Algorithm 5: MethodTap: Packet Tapping

Input: *Pointer to packet *p*
*Initialisation : Declare Struct hdr → fetch header info of *p*
hrd → { uid = p.uid, orgBy = p.fwrdId, src =p.src, dest = p.dest }
var pb → pstore.packetLookUp(hdr.uid)
if *pb != null & (pb.packetId == hdr.uid) & (pb.source == hdr.src) & (pb.dest == hdr.dest) &(pb.fwdrId == hdr.orgby)* **then**
    call *pstore.deletePacket(hrd.uid) → to delete the p* from cache*
    invoke $UpdateTrustScore(hdr.orgby)$
else
    Do nothing
end
Output: Invokes $UpdateTrustScore(hdr.orgby)$

Algorithm 6: UpdateTrustScore: Updating Trust Score

```
1 Input:header struct hdr for each packet *p
2 trust → tstore.trustLookUp(hdr.orgBy)
3 if trust ≤1 then
4 |     tstore.trustUpdate(hdr.orgBy, trust+0.001)
5 else
6 |  tstore.trustUpdate(hdr.orgby, 1)
7 end
8 Output: Trust score for a node is updated
```

Algorithm 7: checkMalicious: Checking Malicious Node

```
1  Input: header struct hdr for each packet *p
2  if (pstore.packetCount ≥ 50) then
3  |     trust_orgBy → tstore.trustLookUp(orgBy)
4  |  if trust_orgBy then
5  |  |     tstore.trustUpdate(orgBy,0)
6  |  else
7  |  |  tstore.trustInsert(orgBy,0)
8  |  end
9  else
10 end
11 Output: TrustScore for a node is updated
```

6. **Trust Value Calculation** Sc: Every node which receives RREP from its neighbor in a network determines the trust value that represents the trustworthiness of the neighboring node. This is the first level of trust score calculation that happens in our proposed model. The trust value Sc of node $M1$ is calculated dynamically based on the number of packet forwarded N^{fwd} by node M_1.

   As shown in pseudo-code 5 for every matched packet that goes out of node M_1, the trust score is incremented by 0.001 by invoking UpdateTrustScore method. However, if the accumulated packet in the pstore is greater than the threshold (λ) at any point in time during the transmission, the trust score Sc is decremented to 0.0 as shown in algorithm 6 by checkMalicious method. The threshold (λ)is a dynamically set value based on the criticality of the message transmitted via V2V. For instance, the threshold value for life-critical message dissemination such as Emergency Electronic Brake Light (EEBL) or Collision Avoidance can be set low so that no packets can be stacked in the queue for a long time. For our experiments, we have considered a threshold value of 50 as the maximum number of packets accumulated in the packet store. This is the total number of packets sent out in 2.5s at the rate of 1 per 0.05s during the transmission simulation. Once the set threshold is reached, the trust value is decremented to 0.0. This approach helps us to rightly identify the fake node that tries to deceive the system.

   Consider a case in which a node V_i is a busy intermediate node for many communications. This node might try to deceive the system by transmitting packets until it achieves higher trust scores. After that, it either drops the packets intentionally or is being controlled by an adversary. In this case, decrementing the trust score by some percentage would still consider the node as honest since

it is not blacklisted yet. Consequently, this results in a malicious node being considered trustworthy. Hence, to overcome this situation, we immediately drop the trust score to 0.0 once the node is identified as dropping packets in our design. However, the threshold value can be set high for noncritical events, and it can be set by using learning algorithms [50]. If no packets are dropped, and the number of packets in pstore is below the threshold, the trust value Sc is incremented until it reaches 1.0. Once the trust score is calculated, node $H3$ uploads a transaction containing trust scores to the nearest RSU.

Algorithm 8: Creating Vehicular Nodes and Ratings List

Input: $T_x \rightarrow$ Total Number of transaction for 120s in a pool
initialize $V_x \rightarrow$ arrayList of all vehicular ids
$r_x \rightarrow$ arrayList of all vehicular ratings
for $T_i \in T_x$ **do**
  push $V_i \longleftarrow$ to V_x
  push $r_i \longleftarrow$ to r_x
end
Output: V_x and r_x

Algorithm 9: Creating Multimaps of Flattened Trust Scores

Input: $V_x \rightarrow$ arrayList of vehicular ids
$r_x \rightarrow$ arrayList of vehicular ratings
create a multimap with vehicle id as key and multiple ratings as value.
for $i \leq V_i$ *size* **do**
  put V_i into *nodes* multimap and map it with multiple r_i
end
Output: *nodes* $\rightarrow$ Map of nodes : corresponding multiple trust scores

7. **Transaction Logging**: In our model, vehicular node V_i invokes an object of transaction class and posts an updated trust score of monitored neighboring vehicle V_j as a transaction in the distributed storage maintained by RSUs. Every transaction must be signed by a digital signature $Sig(SK(V_i))$ of the initiating entity. A typical transaction is represented in table 6.2 below. Each

Algorithm 10: Aggregating Trust Scores

Input: *nodes* → multimap of nodes : corresponding trust scores
avg_{ri} → Average rating for V_i after aggregation
for $node_i \in nodes\ set$ **do**
    Create r_i list
    for *every rating* $r \in r_i$ *list* **do**
        Aggregate and Calculate Average avg_{ri}
    end
end
Output: avg_{ri} → Average rating for V_i after aggregation

Algorithm 11: Blacklisting Nodes

Input: *nodes* → List of all nodes
avg_{ri} → Average rating for V_i after aggregation
for $node \in nodes\ set$ **do**
    if $avg_{ri} \leq threshold$ **then**
        nodes.remove(node)
    else
        Do nothing
    end
end
Output: *nodes* → mutated list of active nodes

transaction is uniquely identified by a transaction id represented by Tx_{id}.

Table 6.1: Typical Transaction by a Member Node

Tx_{id}	$ACRAF23DB3C4$
TimeStamp	$YYYY-MM-DDTHH:MM:SS$
SourceNodeId	PubKey(V_i)
NeighbourNodeId	PubKey(V_j)
TrustScore	$0.0-1.0$
Hopcount	n
digital signature	$Sig(SecKey_{vi})$
Transaction Hash	$Hash(Tx)$

8. **Transaction Pool Processing**: Over time, unconfirmed transactions created by the vehicular nodes throughout the network get accumulated in the distributed ledger transaction pool. This processing of the transaction pool starts with algorithm 7, where the system reads each transaction T_x from the pool and creates a list of vehicular nodes V_x and its corresponding list of ratings r_x. It is further processed as shown in pseudo-code 8, where each unique V_x is mapped to multiple ratings associated with it. Once the transaction pool is processed, it outputs a list of vehicular nodes and their corresponding trust scores. This enables the authorized validators, which are RSUs in our model, to get an accurate list of recent transactions that have to be added to the block.

9. **Leader RSU Selection**: We use Practical Byzantine Fault Tolerance (PBFT) as the underlying consensus protocol. Developed by Castro and Liskov [38] in 1999, PBFT has gained wide recognition for practicality. In our system, authorized RSUs are the validator nodes that follow PBFT protocols to generate and broadcast blocks. One of the RSUs is randomly chosen as the primary or the leader node, and others are secondary.

The leader node collects all received records of transactions and generates a Merkle hash value of the records linked to the previous block in the vehicular blockchain and successfully create a block. Once a leader makes a block, it is validated by the secondary nodes, and all honest nodes help reach a consensus regarding the state of the system using the majority rule. A PBFT enabled distributed system provides a practical byzantine state machine replication that can work even when malicious RSUs are operating in the system, assuming that honest RSUs are more than $2f + 1$ where f is the number of faulty RSUs.

10. **Trust Score Aggregation by RSUs**: There is a possibility that several neighbor nodes could verify each node. In our e.g. $H3$ and $H2$ both nodes validated $M1$. Hence ratings have to be aggregated before creating blocks. Leader RSU node picks all the logged transactions from the processed transaction pool in every defined time interval. It aggregates every vehicular node's ratings based on Algorithm 9. Algorithm 10 provides the pseudo-code for blacklist node table generation. Further to this, it also invokes a Merkle tree module to create a Merkle root hash of all the transactions.

11. **Block Generation and Addition to Blockchain** : Once the trust scores are aggregated, and the blacklisted node table is generated, it is added to the block along with the previous block hash and Merkle root. The created block is pushed for verification from secondary nodes. They validate the correctness of the block and send approval messages to the leader node. After receiving 2/3rd approval, the leader node adds the block to the blockchain and notifies all the nodes as per the PBFT protocol.

12. **Download Blacklist Node Table**: All RSUs update their local chain with the latest one to reflect the latest transactions. All vehicular nodes update

the local database with the updated blacklisted node table and the highest DSN recorded. Source nodes waiting to deliver messages confirms whether the intermediate node responding with the RREP message is blacklisted or not. If yes, it sends out fresh RREQ. Else, communication packets are sent via the found root.

13. **Public Key Revocation and Reactivation**: Once a vehicular node V_i is blacklisted, its associated identity vector $\{id_{vi}, PK_{vi}, SK_{vi}, RC_{vi}\}$ will be revoked before its intended expiration date. A node can retry at most 3 times to re-register in the network. RSU checks the license plate number and fetches the RC_{vi} count for the node V_i requesting re-registration. If RC_{vi} is below 3, the request for a new unique identity vector containing a unique public key PK_{vi} is issued. This gives vehicular nodes a fair amount of chance to rejoin and correct the misbehavior.

6.5 Security Analysis

In this section, we discuss the security features of our proposed model framework. Specifically, this analysis is focused on the resilience against the attacks discussed in section 3.2. Table 3 below summarizes a comparative analysis of security features of the various other approaches discussed in section 2.

6.5.1 Defense against Sybil Attacks

As we have discussed in the system initialization phase of the proposed framework, which is responsible for validating the vehicular node's identity and creating a unique passphrase. From these passphrases, unique identities are generated when nodes move

Table 6.2: Comparative Security Analysis Of The Existing Trust Models For VANET

Secured Against	[32]	[30]	[31]	[33]	[34]	[13]	[35]	OurWork
UnAuthorized Identity	✓	NA	NA	NA				✓
ID Spoofing						✓		✓
Defaming						✓		✓
Byzantine RSU						✓		✓
Data Tampering						✓	✓	✓

into a network. During the transaction logging phase, each transaction T_j needs to be signed by the current vehicle node $DigSign_{id_{vi}}$ before they are sent. The authorized RSU then verifies the signature. Consortium blockchain, combined with the digital signature technique, ensures that any external attacker cannot disrupt the network as the attacker's digital signature cannot be verified. Additionally, an external attacker cannot launch an identity spoofing attack as no entity can falsify the digital signature of another entity without the private key of the actual member of the network. This, in turn, ensures that only legitimate and authenticated vehicles can upload the trust scores for the network entities.

6.5.2 Defense against Defaming Attack

A malicious vehicular node (different from a blackhole node) may evaluate the honest neighbor node to calculate the trust score. It may upload a fake trust score to eliminate the honest behaving node from the network. However, the proposed scheme is secure against the defaming attack. In our model, this attack is defended in two

ways. Firstly, each node can submit trust score once for a specific neighbour node and upload the transaction tuple T_j=$Src_{vid}|Node_{vjd}|TrustScore_{vjd}|Hop|DigSign_{vid}$. Duplicate tuples with the same Src_{vid} and $Node_{vjd}$ are eliminated by RSU's. Secondly, trust score for specific node T_j is aggregated by RSUs by averaging over multiple m records of trust score i.e. $\frac{1}{m}\sum_{x=1}^{m} T_j$ as reported from different neighbour nodes. Lastly, due to the limited number of malicious nodes, these unfair trust scores can hardly disrupt the system.

6.5.3 Security against Tampering Blacklisted Node table

Tamper-proof is another essential feature of this framework. Since the blacklisted node table is distributed in a decentralized manner via blockchain, it is free from any internal or external entity performing add, delete, or modification to the blacklisted node list. This is because of the inherent properties of blockchain; any changes made to the stored node table will inevitably change the hash value of the block resulting in the mismatch and invalidation of the block by the majority of the honest nodes.

6.5.4 Defense against Byzantine RSUs

The proposed system discussed that a small portion of RSUs might get controlled by an attacker. Data might get altered or deleted by these malicious RSUs. However, the PBFT consensus mechanism employed in the system ensures the network's regular operation even when 33% of the nodes are damaged. In PBFT consensus mechanisms, the block proposer is bound to get at least $\frac{2}{3}$rd of votes from the secondary RSUs that are honest. If we suppose that there are f malicious RSU nodes in the whole network and the total number of RSUs satisfies $n \geq 3f + 1$, the system can defend against malicious tampering data attacks initiated by faulty RSU. This makes our

system byzantine fault-tolerant reducing the impacts of compromised RSUs.

6.6 Experimental Evaluation

6.6.1 VANET Simulation Setup

To study the impact of insider attack scenarios and test the network performance with the proposed blockchain-based trust score management solution in VANET, we utilize the NS2 simulation tool installed on Virtual Linux OS with Ubuntu 16.04 distribution having 8.00GB RAM.

In our study for the simulation of real-time roads, junctions, and traffic light, we used OpenStreetMaps (OSM)[1], which provides free editable maps of the world. OSM helps generate realistic street structures considering two-way, four-way streets, traffic lights, and buildings. For vehicular mobility, we use the Simulation of Urban Mobility tool (SUMO) [51] version 1.23. The generated traffic models are then imported into NS2 to simulate various attack scenarios. We restrict the simulation area to 800x800m and repeat evaluation for ten iterations with 20, 40, 60, 80, and 100 vehicular nodes each time. Since the area under simulation is quite smaller, we test with a maximum of 100 nodes for our experiments. The simulations generate trace files analyzed using AWK scripts to calculate average packet delivery ratio (PDR), average throughput, and average delay when the messages are communicated in a network.

We used the TwoRayGround propagation model with a maximum speed of 15m/s. WirelessPhy was used as the network interface type in the configuration file. The UDP traffic was used to send data from source to destination nodes faster as it does not require a 3-way handshake to establish a connection. Packets among the nodes were transmitted with a constant bit rate (CBR) of one packet per 0.05 second. We used

[1]http://www.openstreetmap.org

a constant size of 512 bytes for each packet for all our simulations.

We also vary the number of sources and destination nodes as two for 20, three for 40 and 60, four for 80, and 100 number of total nodes. These nodes sent and received data packets throughout the simulation. The simulation was done for 120 seconds (2 mins).

6.6.2 Blockchain Simulation Setup & Analysis

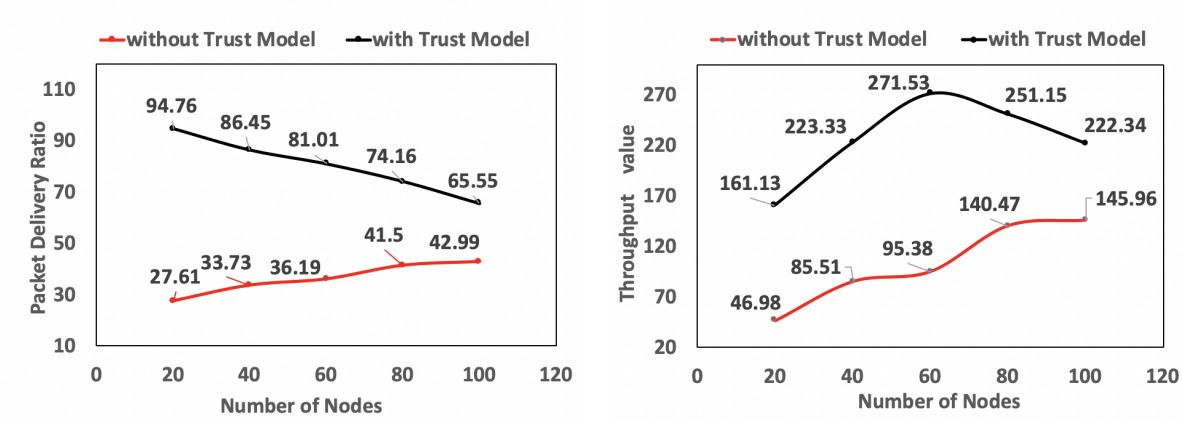
We have implemented trust score management in the Java environment using a laptop with 2.3 GHz Intel Core i5 and 8 GB 2133 MHz LPDDR3. Our simulated blockchain framework receives trust scores data from vehicular nodes through V2I communication for the trust score aggregation. We consider that a network containing N nodes requires at least 15% of maximum network strength as validator nodes. Hence we perform all the tests with 15 validator nodes, i.e., RSUs. We implement various methods to calculate each node's aggregated trust scores, which are disseminated in the blockchain network along with the list of blacklisted nodes.

We mainly consider the following different scenarios.

1. Impact on network performance using trust in AODV routing protocol of VANET, which consists of approximately 25% of malicious nodes.

2. Computation cost of transaction pool and trust score aggregation logic, and derive insights on the scalability of the proposed trust score system.

3. Analyze the total time taken to create a block with 15 entities as validators using PBFT consensus.



(a) (b)

Figure 6.5: Variation of network performance of VANET nodes with and without trust model

6.6.3 Influence of incorporating trust in AODV on network performance

To analyze if the network performance could be improved by incorporating the trust in AODV routing protocol, we ran the simulation twice under the same configuration by injecting 25% malicious nodes. First, we ran it with traditional AODV protocol, and then we ran the proposed AODV with trust model. These malicious nodes could be single blackhole nodes causing the packet drop or multiple nodes forming tunnels to consume the data packets. In the first case, without trust and having 25% of malicious nodes, it is clear that the network throughput in bps and packet delivery ratio significantly deteriorated, as shown in Fig. 6.5(a). From Fig. 6.5(b), it can be inferred that a significant reduction in packet drop ratio and improved throughput in bits per second can be achieved by incorporating the proposed model into the VANET routing protocol. Furthermore, it is evident that when the number of nodes is 40 and 60 with the 25% malicious nodes, the packet drop ratio is of the same range as we are testing with two pairs of source and destination. Similar is the case with 80 and 100 nodes.

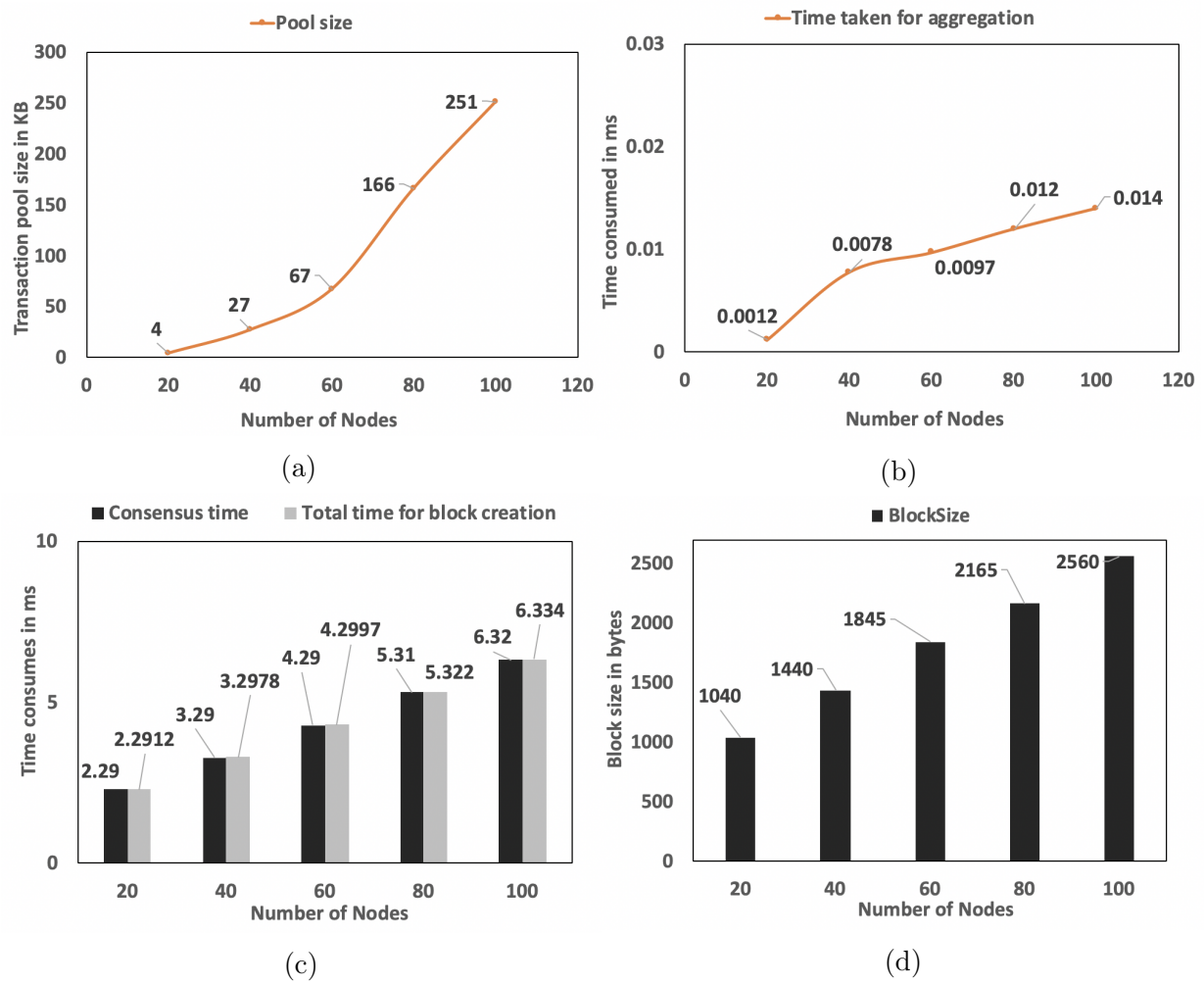


Figure 6.6: Analysing the data metrics in blockchain (a) Size of the transaction pool created with different number of nodes (b) Time consumption of transaction pool processing and block creation vs the number of network nodes

6.6.4 Computation cost of transaction pool processing and trust score aggregation

In this experiment, we have used the data generated from the simulation of Trusted AODV protocol (NS2 tool) as an input to the trust management prototype built based on simulated blockchain. Trust scores data generated as the simulation output was used for logging as transactions in the transaction pool. This data was processed as per Algorithm 7 to 10, and the time consumption for processing various sizes of data pool was evaluated. Different configurations in NS2, i.e., 20, 40, 60, 80, and 100 nodes generated 4KB, 27KB, 67KB, 166KB, and 251KB size of trust scores as transactions in the transaction pool, respectively, as shown in Fig. 6.6(a). Time taken by the validator nodes, i.e., RSUs, to process these transactions and aggregate the trust scores is the order of few milliseconds, as seen in Fig.6.6(b). Although with the increase in the number of transactions, time consumed to aggregate changes linearly, we infer that with the highest of 100 vehicular nodes, the maximum transaction pool size of 251KB was processed, and aggregated trust scores were calculated in as low as 0.014ms.

6.6.5 Computation cost of block creation using PBFT consensus using a varied number of validators

In this experiment, we evaluate the proposed model in terms of time taken for reaching PBFT consensus among validators on the state of transactions and then creating a block with a blacklisted node table. We perform this test with 15 validator nodes, i.e., RSUs, to process different block sizes. Fig 6.6(c) represents the time taken to reach consensus and the total time consumed for each block creation. Block creation with a maximum of 15 validators for a network size of a maximum of 100 vehicular

nodes is taking 6.334 seconds to create a new block. This shows that blacklisted node tables are disseminated to all the network participants within a short duration. Additionally, a block is being created every $120s$ with an average size of $1.8KB$. Fig. 6.6(d) represents the blocksize with a different number of network nodes. Storage overhead is calculated to be around $1.8 * (60/120) * 24 * 365$, which is approximately 7884KB per year for the entire blockchain. Hence, the techniques introduced also require very low storage in the blockchain for the 800x800 area considered in the experiment.

6.7 Summary

In this paper, a consortium blockchain-based approach for mitigating insider attack in the VANET system using Trusted AODV protocol is proposed. We used promiscuous mode to assign a trust value to neighbor vehicular nodes that responded dynamically with RREP messages. The results showed an improved packet delivery ratio and throughput of the entire network by incorporating trust in the AODV routing protocol of VANET. To efficiently distribute trust scores and to eliminate blackhole nodes from the network, we designed a blockchain-based trust score management system. In this design, we also showed how the trust score gets aggregated by authorized RSUs. The vehicular nodes offloaded the mining process to the RSUs to speed up the block generation, suitable for the proposed VANET system. We evaluated the block time consumption concerning PBFT consensus and trust score aggregation. Results were presented to demonstrate that it would be an efficient system for trust score dissemination and is very efficient in eliminating the blackhole nodes in the VANET.

Chapter 7

Limitations, Future Work & Conclusion

7.1 Limitation

In chapter 4, we have presented our preliminary work, PEBERS - A ride-sharing application for VANET based on a public blockchain. We then extended it to include a filtering strategy for miner node selection based on Proof-of-driving and service score for PBFT consensus mechanism in chapter 5. This work is novel yet has some limitations. This system is assumed to contain more than two-third of honest nodes. A possible improvement would be to design specific countermeasures for a 51% attack that is probable in such a network. In chapter 6, The performance of the proposed trust management system design was analyzed via simulation using NS2 for an area of 800x800 with a maximum of 100 nodes in terms of PDR and packet loss. The simulation showed a significant improvement in network metrics with the incorporation of trust with this range. Nevertheless, a drawback of this design is the area-based

blockchain network of vehicles. The scalability and the time consumed for trust calculation and its dissemination is ensured in this system by implementing a local blockchain that is independent of different geographical regions. In a highly dynamic and high-speed vehicular network, where vehicles enter and leave an area quickly, inter blockchain network communication must be established.

7.2 Conclusion

In this thesis, we have demonstrated a case study of designing and developing a specific application of a vehicular network, i.e., ride-sharing based on blockchain. We mainly focused on how blockchain technology for ride-hailing services combined with smart contracts can be implemented. The crucial feature is our smart contract deployed in the network, which helps passengers join the rides and brings trust in the system. As a further study in our following chapters, we proposed an efficient and effective miner node selection strategy for a VANET application under investigation. The selection algorithm is designed to be randomized yet without compromising on the quality of the mining nodes. Results presented proved the effectiveness of the proposed methods. We also presented our study in the direction of VANET internal attacks such as Blackhole attacks. In an attempt to deliver a promising trust score management system, we designed and developed a consortium blockchain-based trust management system for a trusted AODV protocol-based VANET. Results were presented to demonstrate that it would be an efficient system for trust score dissemination and is very efficient in eliminating the black hole nodes in the VANET.

7.3 Future Work

In the future, the proposed blockchain-based system for a ride-sharing application can be integrated with a front end and deployed on a real network for usage. The proposed mining strategies can also be applied for other applications of VANET that consist of an extensive network. Apart from that, various other distributed ledger technology platforms can be leveraged to implement the blockchain-based system. Finally, the trust management system for a larger area with intercommunication between different local blockchains can be researched. The proposed approach can also be modified to use various applications containing data aggregation and distribution.

Bibliography

[1] S. Kudva, S. Badsha, S. Sengupta, I. Khalil, and A. Zomaya, "Towards secure and practical consensus for blockchain based vanet," *Information Sciences*, vol. 545, pp. 170–187, 2021.

[2] M. B. Mollah, J. Zhao, D. Niyato, Y. L. Guan, C. Yuen, S. Sun, K.-Y. Lam, and L. H. Koh, "Blockchain for the internet of vehicles towards intelligent transportation systems: A survey," *IEEE Internet of Things Journal*, 2020.

[3] P. K. Sharma, S. Y. Moon, and J. H. Park, "Block-vn: A distributed blockchain based vehicular network architecture in smart city." *JIPS*, vol. 13, no. 1, pp. 184–195, 2017.

[4] R. Shrestha, R. Bajracharya, A. P. Shrestha, and S. Y. Nam, "A new-type of blockchain for secure message exchange in vanet," *Digital Communications and Networks*, 2019.

[5] P.-Y. Chen, J.-W. Liu, and W.-T. Chen, "A fuel-saving and pollution-reducing dynamic taxi-sharing protocol in vanets," in *Proc. of the 72nd IEEE Vehicular Technology Conference-Fall*, Ottawa, ON, Canada, 2010, pp. 1–5.

[6] N. Liu, M. Liu, J. Cao, G. Chen, and W. Lou, "When transportation meets communication: V2p over vanets," in *Proc. of the 30th IEEE International Conference on Distributed Computing Systems*, Genova, Italy, 2010, pp. 567–576.

[7] K. J. O'Dwyer and D. Malone, "Bitcoin mining and its energy footprint," in *Proc. of the 25th IET Irish Signals Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communications Technologies (ISSC 2014/CIICT 2014)*, Limerick, Ireland, 2014, pp. 280–285.

[8] F. Sakiz and S. Sen, "A survey of attacks and detection mechanisms on intelligent transportation systems: Vanets and iov," *Ad Hoc Networks*, vol. 61, pp. 33–50, 2017.

[9] S. Underwood, "Blockchain beyond bitcoin," *Communications of the ACM*, vol. 59, no. 11, pp. 15–17, 2016.

[10] R. Shivers, M. A. Rahman, and H. Shahriar, "Toward a secure and decentralized blockchain-based ride-hailing platform for autonomous vehicles," *arXiv preprint arXiv:1910.00715*, 2019.

[11] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. of the IEEE International Congress on Big Data*, Honolulu, HI, USA, 2017, pp. 557–564.

[12] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, 2012.

[13] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495–1505, 2018.

[14] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proc. of the Third Symposium on Operating Systems Design and Implementation*. New Orleans, Louisiana, USA: USENIX Association, 1999, p. 173–186.

[15] M. Asghari, D. Deng, C. Shahabi, U. Demiryurek, and Y. Li, "Price-aware real-time ride-sharing at scale: an auction-based approach," in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2016, p. 3.

[16] S. Khanji and S. Assaf, "Boosting ridesharing efficiency through blockchain: Greenride application case study," in *2019 10th International Conference on Information and Communication Systems (ICICS)*. IEEE, 2019, pp. 224–229.

[17] D. Sánchez, S. Martínez, and J. Domingo-Ferrer, "Co-utile p2p ridesharing via decentralization and reputation management," *Transportation Research Part C: Emerging Technologies*, vol. 73, pp. 147–166, 2016.

[18] A. Pham, I. Dacosta, G. Endignoux, J. R. T. Pastoriza, K. Huguenin, and J.-P. Hubaux, "Oride: A privacy-preserving yet accountable ride-hailing service," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 1235–1252.

[19] Y. Wu, P. Song, and F. Wang, "Hybrid consensus algorithm optimization: A mathematical method based on pos and pbft and its application in blockchain," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[20] L. Li, J. Liu, L. Cheng, S. Qiu, W. Wang, X. Zhang, and Z. 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.

[21] Y. Yuan and F.-Y. Wang, "Towards blockchain-based intelligent transportation systems," in *Proc. of the IEEE 19th International Conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, 2016, pp. 2663–2668.

[22] P. K. Sharma, N. Kumar, and J. H. Park, "Blockchain-based distributed framework for automotive industry in a smart city," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4197–4205, 2019.

[23] J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, and J. 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.

[24] X. Liu, W. Wang, D. Niyato, N. Zhao, and P. Wang, "Evolutionary game for mining pool selection in blockchain networks," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 760–763, 2018.

[25] J. Kang, Z. Xiong, D. Niyato, P. Wang, D. Ye, and D. I. Kim, "Incentivizing consensus propagation in proof-of-stake based consortium blockchain networks," *IEEE Wireless Communications Letters*, vol. 8, no. 1, pp. 157–160, 2018.

[26] K. Peterson, R. Deeduvanu, P. Kanjamala, and K. Boles, "A blockchain-based approach to health information exchange networks," in *Proc. of the NIST Workshop Blockchain Healthcare*, vol. 1, 2016, pp. 1–10.

[27] J. Yu, D. Kozhaya, J. Decouchant, and P. Esteves-Verissimo, "Repucoin: Your reputation is your power," *IEEE Transactions on Computers*, vol. 68, no. 8, pp. 1225–1237, 2019.

[28] S. Wang, X. Huang, R. Yu, Y. Zhang, and E. Hossain, "Permissioned blockchain for efficient and secure resource sharing in vehicular edge computing," 2019.

[29] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proc. of the 26th ACM Symposium on Operating Systems Principles*, Shanghai, China, 2017, pp. 51–68.

[30] H. Almutairi, S. Chelloug, H. Alqarni, R. Aljaber, A. Alshehri, and D. Alotaish, "A new black hole detection scheme for vanets," in *Proceedings of the 6th International Conference on Management of Emergent Digital EcoSystems*, 2014, pp. 133–138.

[31] K. M. A. Alheeti, A. Gruebler, and K. D. McDonald-Maier, "An intrusion detection system against black hole attacks on the communication network of self-driving cars," in *2015 sixth international conference on emerging security technologies (EST)*. IEEE, 2015, pp. 86–91.

[32] Z. Lu, Q. Wang, G. Qu, and Z. 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)*. IEEE, 2018, pp. 98–103.

[33] N. Rafique, M. A. Khan, N. A. Saqib, F. Bashir, C. Beard, and Z. Li, "Black hole prevention in vanets using trust management and fuzzy logic analyzer," *International Journal of Computer Science and Information Security*, vol. 14, no. 9, p. 1226, 2016.

[34] Y. Khamayseh, A. Bader, W. Mardini, and M. B. Yasein, "A new protocol for detecting black hole nodes in ad hoc networks," *International Journal of Communication Networks and Information Security*, vol. 3, no. 1, p. 36, 2011.

[35] J. Tobin, C. Thorpe, and L. Murphy, "An approach to mitigate black hole attacks on vehicular wireless networks," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*. IEEE, 2017, pp. 1–7.

[36] L. Zhu, Y. Wu, K. Gai, and K.-K. R. Choo, "Controllable and trustworthy blockchain-based cloud data management," *Future Generation Computer Systems*, vol. 91, pp. 527–535, 2019.

[37] S. Kudva, R. Norderhaug, S. Badsha, S. Sengupta, and A. Kayes, "Pebers: Practical ethereum blockchain based efficient ride hailing service," in *Proc. of the IEEE International Conference on Informatics, IoT and Enabling Technologies*, Doha, Qatar, 2020, pp. 422–428.

[38] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Transactions on Computer Systems (TOCS)*, vol. 20, no. 4, pp. 398–461, 2002.

[39] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," http://bitcoin.org/bitcoin.pdf," 2008.

[40] C. Perkins, E. Belding-Royer, and S. Das, "Rfc3561: Ad hoc on-demand distance vector (aodv) routing," 2003.

[41] P. S. Gautham and R. Shanmughasundaram, "Detection and isolation of black hole in vanet," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*. IEEE, 2017, pp. 1534–1539.

[42] U. Khan, S. Agrawal, and S. Silakari, "Detection of malicious nodes (dmn) in vehicular ad-hoc networks," *Procedia computer science*, vol. 46, no. 9, pp. 965–972, 2015.

[43] M. S. Sheikh and J. Liang, "A comprehensive survey on vanet security services in traffic management system," *Wireless Communications and Mobile Computing*, vol. 2019, 2019.

[44] S. Kudva, S. Badsha, S. Sengupta, H. La, I. Khalil, and M. Atiquzzaman, "A scalable blockchain based trust management in vanet routing protocol," *Journal of Parallel and Distributed Computing*, vol. 152, pp. 144–156, 2021.

[45] X. Zhang and X. Chen, "Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network," *IEEE Access*, vol. 7, pp. 58 241–58 254, 2019.

[46] R. Shrestha and S. Y. Nam, "Regional blockchain for vehicular networks to prevent 51% attacks," *IEEE Access*, vol. 7, pp. 95 021–95 033, 2019.

[47] D. Germanus, S. Roos, T. Strufe, and N. Suri, "Mitigating eclipse attacks in peer-to-peer networks," in *2014 IEEE Conference on Communications and Network Security*. IEEE, 2014, pp. 400–408.

[48] F. Thachil and K. Shet, "A trust based approach for aodv protocol to mitigate black hole attack in manet," in *2012 International Conference on Computing Sciences*. IEEE, 2012, pp. 281–285.

[49] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Conference on the theory and application of cryptographic techniques*. Springer, 1987, pp. 369–378.

[50] N. Taherkhani and S. Pierre, "Centralized and localized data congestion control strategy for vehicular ad hoc networks using a machine learning clustering algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3275–3285, 2016.

[51] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind, 2011.