9-12-2024

Proof-of-Presence: A Blockchain Based Traffic Data Collection System

Jiasong Liu
University of Windsor

Proof-of-Presence: A Blockchain Based Traffic Data Collection System

By

Jiasong Liu

A Thesis
Submitted to the Faculty of Graduate Studies
through the Department of Electrical and Computer Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science
at the University of Windsor

Windsor, Ontario, Canada

2024

Proof-of-Presence: A Blockchain Based Traffic Data Collection System

by

Jiasong Liu

APPROVED BY:

S. Jiang
School of Computer Science

S. Chowdhury
Department of Electrical and Computer Engineering

N. Zhang, Advisor
Department of Electrical and Computer Engineering

August 27, 2024

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Traffic data collection plays an essential role in the operation of connected autonomous vehicle (CAV). The vehicles rely on authentic, accurate and immediate data to make informed decisions about the most efficient and safest routes for delivering passengers. Basic safety messages (BSM) can serve as a source for traffic data collection, as they contain comprehensive data from CAVs. Also, the rapid generation rate of the BSMs ensures the timely of the relevant traffic data. However, given the design of BSM. They are susceptible to various types of attacks, which can be easy to execute. In this thesis, we have developed a method to ensure the authenticity and integrity of BSMs using a decentralized approach: Proof-of-Presence (PoPr). This blockchain system is operates based on proof-of-reputation. The architecture of PoPr is throughly introduced in this thesis, and the simulation results and analysis of the PoPr are provided to illustrate the effectiveness of the system.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor, Professor Ning Zhang for his invaluable support and guidance for this thesis. My appreciation also extends to Jiayuan Wang for assist me with vehicle recognition. Lastly, I would like to thank Fangdi Liu's help in creating Carla image label data.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

3GPP	3rd Generation Partnership Project
AVSN	Autonomous Vehicle Social Network
AU	Application Unit
BSM	Basic Safety Message
BGP	Byzantine Generals' Problem
CAV	Connected Autonomous Vehicle
CA	Central authority
C-V2X	Cellular Vehicle to Everything.
DOS	Denial of Service
DDOS	Distributed Denial of Service
DSA	Digital Signature Algorithm
DSRC	Dedicated Short Range Communication
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Eliptic Curve Digital Signature Algorithm
FCD	Floating Car Data
IPv6	Internet Protocol version 6
ITS	Intelligent Transport System
MANET	Mobile Ad Hoc Network
MitM	Man in the Middle Attack
MPS	Message Per Seconds
GPS	Global Positioning System

OBU	On Board Unit
P2P	Peer-to-Peer
PBFT	Practical Byzantine Fault Tolerance
PoS	Proof-of-Stake
PoPr	Proof-of-Presence
PoW	Proof-of-Work
PKI	Public Key Infrastructure
QFT	Quantum Fourier Transform
RSU	Roadside Unit
UDP	User Datagram Protocol
TCP	Transmission Control Protoco
VANET	Vehicular ad hoc Network
V2I	Vehicle-to-Infrastructure
V2P	Vehicle-to-Pedestrian
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
WAVE	Wireless Access for Vehicular Environments
WSMP	WAVE Short Message Protocol

CHAPTER 1

Introduction

1.1 Traffic Data Collection System

Traffic data collection systems play an essential role to modern transportation management and in the decision-making process of CAVs regarding on route selection. To navigate the most efficient and safest route to their destination, CAVs must rely on accurate, real-time traffic data. Thus, the process of collecting traffic data must ensure that the data collected are accurate and authentic. There are two major ways to collect the traffic data: on-road sensors and Floating Car Data (FCD). The on-road sensors approach can provide high-quality traffic data but comes with high operational and capital costs. Conversely, FCD involves collecting real-time traffic data by tracking vehicles using mobile phones or global positioning system (GPS) across the road network [1]. CAVs can be used as a source of traffic data collection since their systems are integrated with GPS technology.

1.2 Vehicular ad hoc Network

A Vehicular ad hoc Network (VANET) is a network designed for moving vehicles and roadside units (RSU). The CAVs function as nodes or routers to exchange messages between vehicles or RSUs. Using dedicated short range communication (DSRC), which utilized 802.11p [2], it can typically connect to CAVs or RSUs within a range of 100 to 900 meters.

VANETs are designed to facilitate both Vehicle-to-Vehicle (V2V) and Vehicle-

to-Infrastructure (V2I) communication within a network lacking infrastructure [3]. In VANET, the CAVs communicate with each other and RSUs using basic safety messages (BSM). CAVs boardcasts BSMs that includes its GPS location, velocity and direction of travel at least once every 100 milliseconds [4]. However, communicating via BSM through DSRC may not be sufficient for sharing traffic data and other resources. Therefore, Cellular Vehicle to Everything (C-V2X) can be used as a complement or enhancement for communication within the VANET. A significant advantage of C-V2X is it offers a larger bandwidth than DSRC. Specifically, the bandwidth allocated for DSRC communication is 75 MHz of licensed spectrum in the 5.9 GHz band [2]. In contrast, C-V2X, utilized NR Uu interface with 5G technology could reach up to 400 Mhz bandwidth in certain modes, although this comes with the tradeoff in power consumption [5]. Given this advantage, CAVs that utilize C-V2X can send and receive larger amounts of traffic data and other types of data with low latency. Figure 1.2.1 shows a sample VANET system.

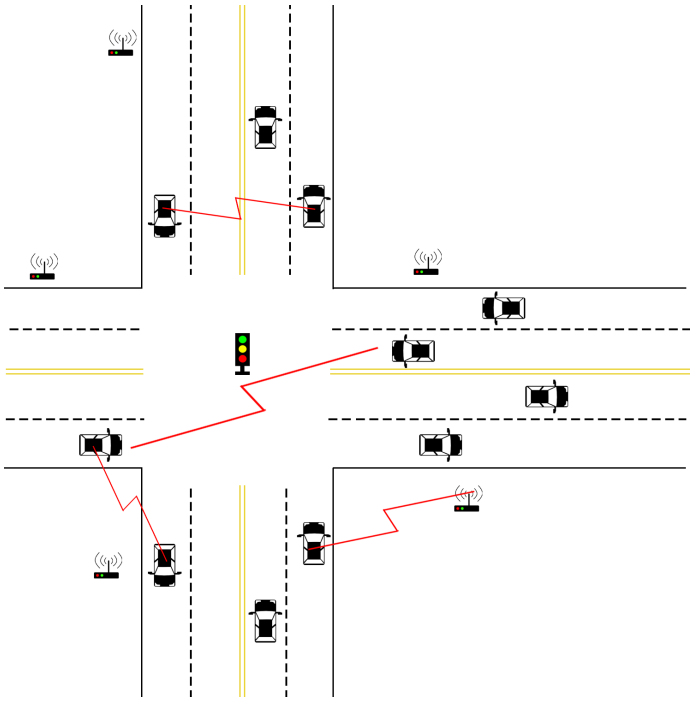

Fig. 1.2.1: A VANET with CAVs and RSUs.

However, the open and dynamic design of VANETs introduces significant security challenges. The traffic data provided by VANETs has a significant impact on the

route planning decisions of CAVs. Therefore, the data from VANETs must satisfy the following characteristics to ensure the safety and integrity of the messages [6]:

- Authentication: Ensures that the message is created by an authenticated user using a certificate, or enables the recipient to recognize the sender of a message via a pseudonym.
- Availability: The network ensures availability by resisting DoS (Denial of Service) attacks, maintaining normal functioning. Even a delay of a few seconds can render the message meaningless.
- Non-repudiation: Ensures neither the sending nor the receiving parties can deny the transmission and reception of data in the event of a dispute [7].
- Integrity: Data remains unaltered, and a digital signature is used to ensure message and data integrity.
- Data verification: Data verification aims to eliminate false messaging by verifying data consistency and validating digital signatures.
- Traceability: While a vehicle's true identity should remain hidden from others, there must be a mechanism in place that can access these real identities to revoke them if needed for future purposes.

To achieve the aforementioned data security goals, one solution is to use Public Key Infrastructure (PKI). PKI is employed as a well-known security standard protocol to enhance the security of the VANET. Figure 1.2.2 shows the architecture of the PKI scheme.

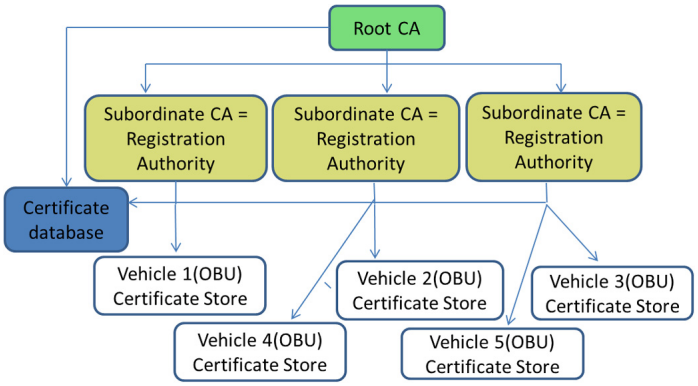

Fig. 1.2.2: PKI model [6].

PKI facilitates the distribution and identification of public encryption keys, allowing users to securely exchange data over the network and verify each other's identities [6]. The PKI approach involves a central authority (CA) playing a critical role in ensuring the safety of VANET. Another approach to tackle this problem is by blockchain. Blockchain refers to a sequential chain of blocks. Each block is responsible for containing transaction information as records and the address of the subsequent block. The header and data within each block are encoded using a hashing mechanism to ensure data integrity [8]. Blockchain operates in a decentralized manner, thereby eliminating the necessity for a central authority (CA) to ensure its functionality. Although some blockchain systems may employ a CA for specific tasks, such as enhancing trust for certain types of nodes.

1.3 Types of Attacks in VANET

Attacks on the VANET can vary based on the design of the VANET system and its hardware. Attacks can be categorized into the following groups: (1) attacks on the wireless interface, (2) attacks targeting hardware and software, (3) attacks on onboard sensors, and (4) attacks on infrastructure systems [6]. Here are some major types of attacks that can target VANETs:

Location Tracking: The attacker will track the driver or CAV by acquiring GPS data through unauthorized methods.

Denial of Service (DoS) or Distributed Denial of Service (DDoS) attack: DoS attacks occur when attackers attempt to overwhelm nodes with malformed requests, rendering the node unavailable to other nodes. DDoS attacks are a type of DoS attack that originate from multiple sources simultaneously.

Spoofing: The attacker attempts to impersonate existing, legitimate nodes in order to deceive other nodes.

Sybil Attack: The attacker creates multiple CAVs with fake identities and attempts to disseminate false or faulty messages.

Man in the Middle Attack (MitM): The attacker intercept communication from both sides of two nodes in the system and pretend to be either one of them to deliver fake or faulty messages.

Brute-force Attack: The attacker will use brute force techniques to crack the target's ID or password and attempt to decrypt the messages. Such attacks typically focus on systems with weak or poorly designed encryption mechanisms.

Message Fabrication and Alternation: The attacker either modifies the content of the message or generate a new faulty messages.

GPS Spoofing: The CAVs generates faulty GPS data intentionally or unintentionally.

Repudiation Attack: The attacker denies the participation of communication or event.

Digital signature Attack: Shor's quantum algorithms for integer factoring and discrete logarithms are approximately equally complex, typically exhibiting a cubic complexity of $O(n^3)$ with quantum computer [9]. The discrete logarithm problem, in particular, serves as the foundational challenge for many widely used digital signature schemes in VANET.

1.4 Blockchain in VANET Security

Introducing blockchain technology to VANET can help address security issues. Blockchain is an emerging decentralized and distributed computing paradigm that underpins the

Bitcoin cryptocurrency, offering privacy and security in Peer-to-Peer (P2P) networks. In the context of VANET, blockchain can be employed to manage information for CAVs, as any vehicle can access the entire history of messages stored on the blockchain [10]. The inherent properties of blockchain ensure that VANET is largely protected from many of the previously mentioned attacks to some extent. Here are the properties of blockchain and how they contribute to the security of VANET:

Decentralization: The blockchain requires no CA to function [11]. This property enhances the security of the blockchain by increasing the difficulty of conducting a DDoS attack. When a CA is required to ensure security, it can become a viable and vulnerable target for DDoS attacks. However, executing a DDoS attack on specific nodes within a blockchain network is challenging because it is difficult to predict which node will extend the block.

Immutability: Immutability is a fundamental characteristic of blockchain technology. Once data is confirmed and attached on the blockchain, it cannot be modified or deleted from the blockchain. Additionally, the new messages to the blockchain is strictly controlled, as they must meet specific conditions before being appended to a new block. The hash values link the blocks, ensuring the blockchain's immutability and tamper-resistance [11].

Anonymity: Blockchain protects nodes' privacy by ensuring that public keys are untraceable to individual users [12]. The anonymity property effectively safeguards the privacy of nodes only when the relationship between the public key and its associated identity is neither intentionally nor unintentionally disclosed.

Non-Repudiation: Every transaction or message is recorded on the blockchain with the signature of the nodes involved. Given the immutable property of the blockchain, it is difficult to deny the occurrence of events or the authenticity of messages once they are logged on the chain.

Consensus Algorithm: The consensus algorithm ensures that nodes on the blockchain will only validate the correct block, where every message or transaction is verified. This is achieved by using the correct block's hash as the new block's ID.

Transparency: The messages stored on the blockchain are fully accessible to

all users. Each block contains comprehensive records of messages and events, offering transparency while maintaining the anonymity of the nodes involved in the blockchain.

Therefore, a VANET system enhanced with a properly implemented blockchain could ensure—or at least partially ensure—the security characteristics of VANET. Authentication and integrity of BSMs are secured using digital signatures. Moreover, the blockchain ensures non-repudiation for every message or transaction, guaranteeing that neither the sender nor the receiver can deny the occurrence of the event. The anonymity property of blockchain guarantees the nodes are unable to trace if their identity is not related with their public key. Most importantly, the blockchain does not require a CA to operate. This allows it to function without a centralized, dominant authority, thereby reducing the risk of a single point of failure due to either faults or DDoS attacks.

However, it is important to note that current popular blockchains like Bitcoin and Ethereum utilize the Elliptic Curve Digital Signature Algorithm (ECDSA), which is based on the underlying difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP) [13]. The security of ECDSA is diminishing with the emergence of quantum computing.

1.5 Post Quantum Digital Signature Scheme

The majority of public key cryptography in use is fundamentally reliant on the presumed computational intractability of integer factorization or discrete logarithms, whether applied to finite fields or elliptic curves. Most blockchains implement the digital signature scheme using ECDSA because it requires smaller key sizes compared to RSA or DSA. This reduction in key size decreases the overhead associated with the size of messages exchanged between nodes. In recent years, quantum computing has seen significant advancements, greatly impacting the efficacy of algorithms like Shor's algorithm. While Shor's algorithm offers limited benefits on classical computers due to its exponential complexity, it achieves a much more manageable cubic

complexity of $O(n^3)$ on quantum computers [9]. As quantum computing progresses, the efficiency and number of qubits are improving, which challenges the security of encryption methods like ECDSA. Decrypting ECDSA's private keys becomes increasingly feasible, compromising its security. Thus, adopting post-quantum digital signature schemes is necessary to secure against the vulnerabilities of ECDSA in the upcoming quantum era. One potential alternative for a digital signature scheme is CRYSTALS-Dilithium, a newer scheme whose security relies on the hardness of the lattice problem [14]. This makes it a viable option for implementation in blockchain systems, where robust security against quantum computing attacks is crucial.

1.6 Motivation and Problem Statement

Building a traffic data collection system that incorporates a VANET with all necessary security characteristics can be challenging but is essential. Traffic data has a significant impact on the route planning decisions of CAVs. Without the guarantee of a secure traffic data collection system, CAVs are vulnerable to various types of attacks, which can lead to compromised performance and suboptimal route choices. Figure 1.6.1 illustrates one possible attack on traffic data and CAV's route planning. The purple lines indicates the fake congestions on the road. The black circle represents the trap location that an adversary aims for the CAV to encounter. The blue line is the route plan made based on the fake traffic data.

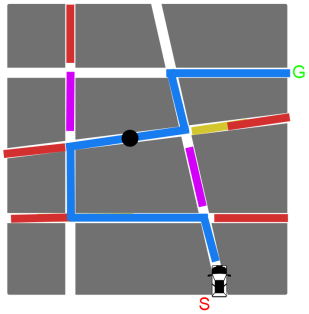

Fig. 1.6.1: One possible attack for traffic data collection system

Conversely, there has been relatively limited research on designing VANET sys-

tems that utilize blockchain technology for traffic data collection. Some research efforts have focused more on aggregating traffic data collected from CAVs or RSUs and sending it to the cloud for validation and processing [15]. Other research efforts place greater emphasis on timing. For instance, one approach designs the data collection process to be initiated by a specific request, specifying details such as the types of data required, along with the maximum allowable duration and depth of the collection [16]. Moreover, due to the initial design of the blockchain consensus algorithm using Proof-of-Work (PoW) [12], the consensus may not be reached in time for effective traffic data collection with a PoW blockchain. Therefore, in current research on VANET blockchain, consensus algorithms are being adapted to accommodate specific or emergency traffic events, ensuring timely and relevant data handling. The Proof-of-Event (PoE) used in VANET blockchain are designed to create new blocks only when a special event is triggered [17, 18]. Therefore, it is challenging to adapt this approach for a general traffic data collection system, as such systems require timely data collection to function effectively. Moreover, the advancement of quantum computing has rendered the widely used digital signature scheme, ECDSA, potentially obsolete. Furthermore, there is a paucity of research on the performance of post-quantum digital signature schemes within blockchain systems, particularly in the context of VANETs, due to their novalty and lack of optimization.

Considering the scarcity of research on using both VANET and blockchain approaches for traffic data collection, this thesis aims to develop a traffic data collection system within VANET that incorporates a specially designed blockchain. This system will facilitate relatively real-time data collection while providing adequate defense measures against various types of attacks targeting both VANET and the blockchain infrastructure.

1.7 Solution Outline

The thesis presented provides a solution for a traffic data collection system using VANET and blockchain technology. Traffic data is collected by analyzing the BSMs

sent by CAVs and RSUs. The system includes both CAVs and RSUs to ensure the traffic data are collected from different sources.

To ensure both the security of VANET and the real-time collection of traffic data, we designed a new blockchain consensus algorithm: Proof-of-Presence (PoPr). The consensus algorithm is based on Proof-of-Reputation (PoR), where each node is assigned a reputation score, and the nodes with the highest scores are grouped together. These grouped nodes are assigned different tasks, such as creating new blocks, to ensure the safety of the blockchain system.

The general structure of PoPr blockchain traffic data collection system is presented as follow:

1. **BSM gathering:** CAVs and RSUs will send BSM to each other by DSRC.

2. **BSM signature verification:** CAVs and RSUs will verify the BSM by the digital signature in BSM with the sender's public key.

3. **Multi-sensor verification:** After the signature's verification is complete, the CAVs and RSUs will match the BSM with the sender who is within the range of their sensors, such as cameras and radars. They will generate verification messages for the BSMs that are able to match with their sensors' data.

4. **Group processing:** CAVs and RSUs will send the verification messages for BSMs to the group members which has high reputation among the nodes.

5. **Block creation:** Group member who selected as leader will create new block contains the BSMs and the verification messages of the BSMs.

6. **Traffic data extraction:** After new block is created, the block is used to update the current traffic condition with the BSMs in the block.

The PoPr blockchain system ensures that the collected traffic data are authentic and accurate. It also guarantees that the system can defend against various VANET attacks, given certain assumptions about the attackers. The overview of the PoPr blockchain consensus procedures is presented in Figure 1.7.1. This figure depicts the

entire process, from the collection of BSM and verification data to the integration of this data into the PoPr blockchain.

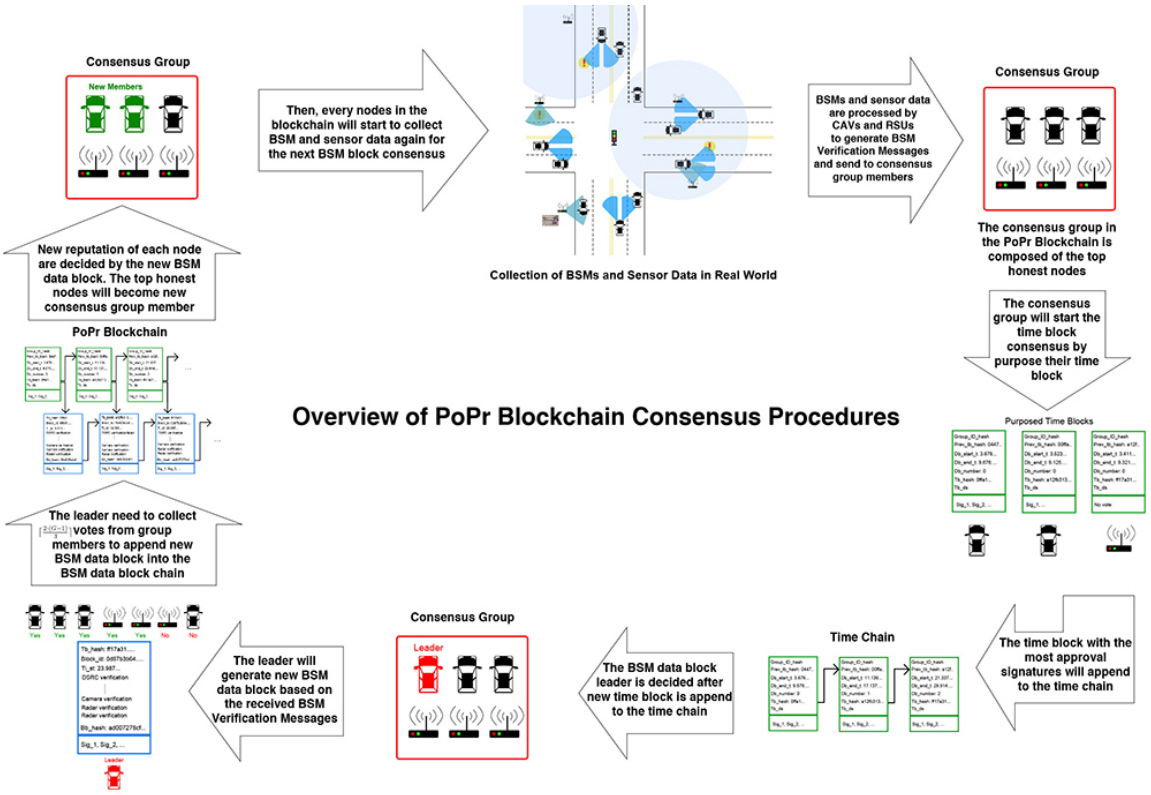

Fig. 1.7.1: Overview of PoPr Blockchain Consensus Procedures

1.8 Thesis Organization

The following thesis is organized as follows: In Chapter 2, we present the overview of traffic data collection system, overview of VANET, blockchain application within VANET, consensus algorithms used in blockchain and post quantum digital signature schemes. Chapter 3 focuses on the proposed method and the features of PoPr blockchain to resolve the attacks for VANET. In Chapter 4, we will present the experimental setup for testing the blockchain integration with VANET and analyze the results derived from the experiment. Finally, in Chapter 5 presents with the conclusion and outlines future work to enhance the security of PoPr blockchain-based VANET traffic data collection system.

1.9 Summary

In this chapter, we present an overview of the traffic data collection system, VANET, and the associated security challenges. We also introduce the blockchain system, discuss the limitations of the digital signature algorithm ECDSA, and highlight the need for its enhancement. Finally, we discuss the motivation for this research and outline the structure of the proposed blockchain solution.

CHAPTER 2

Literature Review

2.1 Overview of Traffic data collection system

A traffic data collection system is a crucial component of intelligent transport system (ITS). It is designed to gather, analyze, and utilize information related to vehicular movement, road conditions, and traffic patterns. The system uses conventional on-road sensors and, alternatively, the newer method of Floating Car Data (FCD) to collect real-time traffic data. Conventional methods include [1]:

Piezoelectric sensors: The sensors are installed in grooves along the roadway surface of the monitored lane(s).

Pneumatic road tubes: Rubber tubes are laid across the road lanes to detect vehicles by sensing the pressure variations caused when a vehicle's tire rolls over the tube.

Magnetic loops: The magnetic loops are installed in the roadway in a square configuration, which generates a magnetic field. The received data is then sent to a counting device positioned at the side of the road.

Conventional methods for collecting traffic data are burdened by high costs associated with implementation and maintenance. Consequently, alternative approaches are essential to obtain traffic data in a more timely and cost-effective manner.

In contrast, FCD collects traffic data directly from vehicles using two primary methods: GPS data from mobile phones and GPS integrated into onboard units (OBUs). A common apporach of gathering traffic data from on board unit (OBU) is presented in [19]. The paper explores a system operates on a client-server archi-

tecture, where the clients are vehicles equipped with GPS receivers, and the server is responsible for gathering GPS data from these clients. Using celluar phone without GPS to collect traffic data is also possible. In [20], the paper introduces models that utilize anonymous phone call data to estimate the movement of vehicles between cells. These models incorporate variables reflecting users' calling behavior and other relevant aspects like the hourly intensity of calls and vehicular traffic. However, to develop a traffic data collection system that provides more timely data while ensuring data security, it is necessary to utilize VANET and implement novel security measures.

2.2 Overview of VANET

VANETs are a specialized form of mobile ad hoc networks (MANETs) that enable vehicles to communicate with each other and RSU [3]. By leveraging wireless communication technologies, VANETs facilitate the exchange of critical information, such as traffic condtions, safety warnings, and navigation updates, among CAVs on routes. This real-time data sharing improves driving safety, reduce traffic congestion, and supports the development of ITS. As CAVs become increasingly connected and autonomous, VANETs play a pivotal role in advancing smart mobility and the future of transportation. Having established the importance and foundational aspects of VANET, it is essential to explore the underlying communication models that enable the seamless exchange of information between vehicles and infrastructure.

2.2.1 Communication Models of VANET

In VANETs, the communication models are mainly divided into a few types, including V2V, V2I, vehicle to everything (V2X), and vehicle to pedestrian (V2P), among others. Each type plays a specific role in enabling effective communication within the network. Here are the definitions for some of the communication models:

Vehicle-to-Vehicle (V2V): Vehicles could directly communicate with other vehicles in VANET.

Vehicle-to-Infrastructure (V2I): For communications from vehicles to infrastructure in VANETs, the infrastructure includes RSUs, traffic lights, traffic cameras, traffic controllers, and more.

Vehicle-to-Pedestrian (V2P): Enhances pedestrian safety by enabling communication between vehicles and pedestrians.

Vehicle-to-Infrastructure (V2X): In this communication model, vehicles share messages with any entities, including both V2V and V2I interactions.

V2V and V2I are critical communication models for traffic data collection, as they enable the broadcasting of BSMs and other communications to other CAVs and RSUs. Additionally, the collected traffic data can be uploaded to the internet through V2X technology.

2.2.2 Key Components of VANET

The architecture of VANET comprises several major components designed to enhance communication and safety on the roads. The critical elements for CAVs include Sensors, OBUs, and Application Units (AUs). Additional components like RSUs and cloud-based services also play essential roles. Detailed explanations of these components are provided below [21]:

- Roadside Unit (RSU): RSU are positioned along the roadside at regular intervals for communication. Typically, these units are installed at traffic nodes or parking areas. The role of RSUs are creating a small ad hoc network enhances the communication range by distributing information to various CAVs and RSUs. It can distribute warning messages, traffic data, and more.

- On Board Unit (OBU): The OBU are integrated with CAVs to facilitate communication with other entities. OBUs are crucial for establishing VANETs as they are equipped with wireless communication modules, onboard sensors, and processing units. The device also supports multiple communication standards, allowing it to connect with RSUs that use similar communication protocols.

An OBU contains specific modules that provide functions such as wireless radio access, ad hoc geographical routing, network congestion control, and data security management.

- Sensors: Sensors on OBUs are tailored to detect specific information according to the application design of the VANET. These OBUs are equipped with a set of sensors that enable the transmission of various updates to the CAVs and RSUs. Sensors in an OBU may include onboard cameras, radar, accelerometers, and more, depending on the specific applications of the VANET.

- Application Unit (AU): The AU is typically integrated with the OBU, through which it exclusively communicates. The OBU handles all mobility and networking functions for the AU, facilitating its operation. Additionally, the AU is designed to leverage various services offered by RSUs.

- Electronic Licenses Plate (ELP): ELP acts as a unique identifier for vehicles, enabling them to be located or tracked via RSUs or GPS in cases of accidents or if the vehicle goes missing.

2.2.3 Wireless Standards of VANET

Multiple wireless standards are employed in VANET to facilitate V2V and V2I communications. The protocols for wireless communication in VANETs include Cellular systems, WLAN standards, DSRC/WAVE standards, CALM standards, and others [3]. These wireless protocols can be categorized into several types based on their operational ranges. Figure 2.2.1 illustrates the various categories of protocols, distinguishing them by their range capabilities.

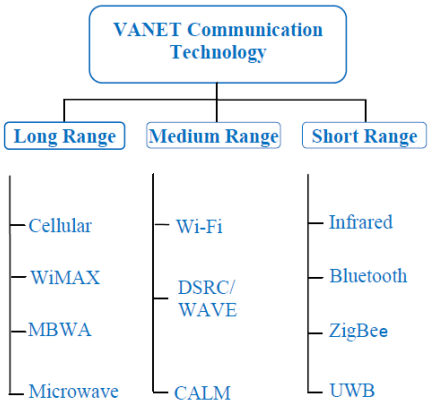

Fig. 2.2.1: Categories of Wireless Protocols Based on Range [3].

DSRC is a medium-range communication protocol specifically developed for VANETs. It is designed to support a self-organized, configured, and dynamically adaptable network topology [3]. DSRC operates within a 75 MHz spectrum at the 5.9 GHz frequency band and is capable of covering a range from 100 meters up to 1 kilometer. This capability makes it well-suited for various operations within VANETs, such as V2V and V2I communications. In Figure 2.2.2, the stack layers of DSRC is presented.

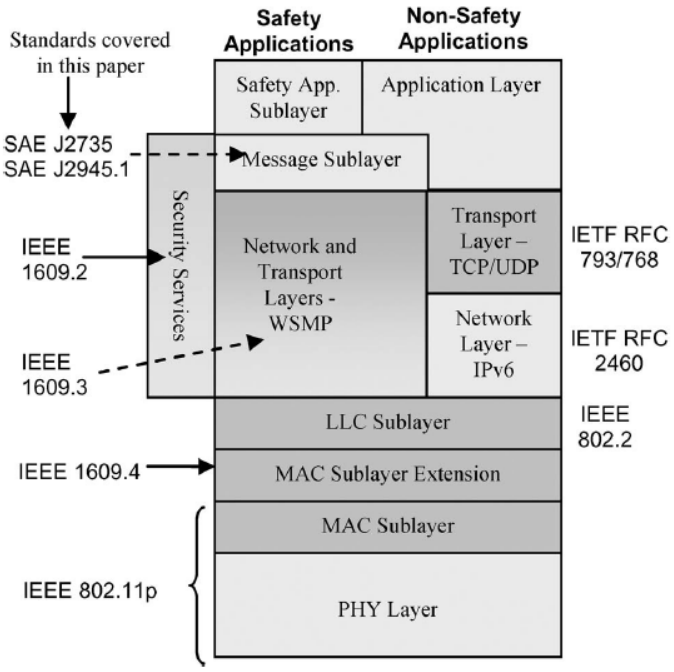

Fig. 2.2.2: DSRC architecture [2].

In the physical layer of DSRC it utilized IEEE 802.11p Wireless Access for Vehicular Environments (WAVE), and this make it a different verison than regular IEEE 802.11p Wi-Fi standard. In the middle layer of the stack, DSRC employs standards defined by IEEE 1609, designed specifically to enhance vehicular communication systems. Additionally, it provides the flexibility to either use the WAVE Short Message Protocol (WSMP) across both the transport and network layers or opt for the Internet Protocol version 6 (IPv6) for the network layer, while supporting both User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) in the transport layer.

Conversely, C-V2X is recognized as a premier long-range communication protocol within VANETs. Introduced by 3rd Generation Partnership Project (3GPP) in 2017, it is designed to bolster vehicular safety and provide essential information services, facilitating seamless vehicle-to-everything interactions. Built on the foundations of Long-Term Evolution and 5G technologies, C-V2X offers both low-bandwidth and higher-bandwidth options, enabling the transmission of more complex data over greater distances compared to DSRC [22]. Moreover, Compared to DSRC, C-V2X offers greater configurational flexibility. It allows for the selection of occupied bandwidth, modulation, and coding schemes at the physical layer, as well as various parameters at the MAC layer [23].

2.3 Blockchain Applications in VANET

The architecture of VANETs makes them prone to several types of attacks, necessitating effective security measures to maintain safe communication within the network.

A feasible approach to building security infrastructures in VANETs is the use of PKI, where a CA plays a central role in verifying the authenticity of messages exchanged between entities. In this approach, the CA serves as the central trust entity, providing services to authenticate the identities of different entities within the VANET. The primary CA can pass certificate issuance to secondary CAs, each tasked with issuing certificates for specific purposes. This hierarchical structure helps to protect the main CA from direct exposure to all nodes in the VANET, thus reducing

the risk of DDoS attacks to some extent. Additionally, CAs maintain a database of valid and invalid certificates, enabling accurate verification of each node's certificates. On the VANET nodes' side, they store and manage certificates issued by the CA, facilitating secure communication among themselves [6].

The PKI approach benefits in terms of efficiency and resource allocation within VANETs. The CA and secondary CAs can issue and verify certificates quickly, relieving VANET nodes from the burden of verification tasks. This is particularly advantageous since CAVs often have limited computational resources. A significant drawback of the PKI system is its vulnerability due to the central role of the CA. If the CA encounters any form of disruption or failure, the entire certificate verification process can be halted, thereby compomising the security of the VANET. Another issue is the privacy concern surrounding the CA. If the security of the CA is compromised, the security of VANET nodes can also be jeopardized. This happens especially when applying for a certificate requires sensitive data from the nodes.

In contrast, blockchain, initially developed by Bitcoin to establish a decentralized cryptocurrency [12], has since diversified into an approach for enhancing the security of various systems. The unique features of decentralization, anonymity, immutability of transactions, and non-repudiation make blockchain a promising approach for enhancing VANET security. Here are few recently researches about blockchain applications to VANET: Guo et al [18] introduced a forensic data recording system for CAV, based on a novel blockchain framework utilizing Proof-of-Event consensus protocol. The system activates when an accident occurs, the nodes in the system are categorizing into "accident vehicles", "witness vehicles" and "community vehicles". Upon an accident, the "accident" vehicle boardcasts a request to witness vehicles and both "accident vehicle" and "witness vehicle" generate the records of the accident, and stored the records to the new block. Moreover, the "accident vehicle" sends requests to a subgroup of "community vehicles" known as "verifier vehicle", to authenticate the accuracy of the record data. Once verification is complete, the leader of "verifier vehicle' initiates a consensus voting process that requires n out of m verifier vehicles to agree. Following the confirmation of the new block by consensus vote,

the leader "verifier vehicle" will boardcasts newly confirmed block to all members of the blockchain. This blockchain is created only in the event of an accident, so the real-time of creating new block data is not a major concern.

In [10], the authors introduced a blockchain for VANETs to enhance the trustworthiness of nodes and message. The scheme utilize trust data a transactions, employing Proof-of-Location issued by RSUs for verifiers' authentication. The authors enhance scalability through local blockchains segmented by geographical regions. Additionally, to decrease latency in new block generation, they proposes integrating edge computing. This method aims to enable real-time applications by offloading complex computations to edge devices. The blockchain system designed for VANETs is lightweight, yet its security relies solely on the authentication provided by RSUs. Conversly, Diallo et al [24] introduced a novel blockchain-based protocol in VANET for secure and reliable management of traffic-related records. The protocol's consensus participants are dynamically selected based on the locations of traffic events, and consensus on blocks is reached using the practical byzantine fault tolerance (PBFT) algorithm. The leader of the blockchain is selected through a round-robin mechanism. This leader is responsible for creating new blocks that adhere to specific conditions, such as a fixed block size and the requirement that all recorded events in the block are within a certain geographical region. Group members, primarily RSUs, are chosen based on their proximity to the location of each event, with those closer having a higher likelihood of being included in the consensus group. Once a new block is created, it is sent to k - 1 nearby RSUs. These RSUs then verify the block and agree to add it to the chain. The blockchain is tested with NS-3 simulator. The approach of using PBFT to collect traffic events helps protect the network from faulty messages. However, it does not mitigate the risk of a 51% attack, where a majority of the group members might already be faulty or controlled by adversaries.

Other blockchain designs for VANETs employ different consensus algorithms, such as using fixed nodes for consensus and implementing a Proof-of-Reputation (PoR) approach to achieve agreement among nodes. In recent studies of consortium blockchain [25], a trust management paradigm for resource sharing in the internet of vehicle, uti-

lizing a consortium blockchain architecture. They argue that a fully decentralized public chain cannot support large-scale networks. In the consortium blockchain architecture, CAVs are excluded from participating in the consensus process. Instead, RSUs are tasked with collecting related transactions and compiling them into blocks of limited size. The reputation of the transactions created by these nodes is determined by the behavior of the CAVs. RSUs prioritize transactions with the highest cumulative reputation, ensuring that the block with the maximum reputation is the one added to the blockchain. The consortium approach is cost-effective but involves compromises in the consensus algorithm. Additionally, using fixed nodes for new block consensus makes them vulnerable targets for DDoS attacks. Another research applied blockchain to secure the content sharing in autonomous vehicle social networks (AVSNs) with PoR [26]. A blockchain framework for AVSNs is introduced, securing content transactions and protecting CAV identity privacy through immutable ledgers and encryption. The reputation assessment models based on nodes' behaviors incentivize legitimate actions and boost content reliability for CAVs and RSUs. The integration of BLS multi-signature and PoR consensus protocols optimizes blockchain deployment in resource-limited AVSNs. In this research, PoW is employed as the consensus method, where the difficulty of the PoW task is adjusted based on the nodes' reputation values. Nodes with higher reputations are assigned easier PoW targets, increasing their likelihood of becoming the consensus node to create a new block. The consensus approach is vulnerable to attacks from collaborating adversarial nodes. Adversaries with high reputations might team up with others who have significant computational power to achieve the PoW target, aiming to take control of the blockchain and compromise the system's security.

In summary, the consensus algorithm is a critical component for ensuring the security of the blockchain system in VANETs. It requires careful examination, and we will introduce several consensus algorithm approaches in the next section.

2.4 Consensus Algorithms of Blockchain

Consensus is a significant component in creation of new blocks within a blockchain. It ensures that all nodes in a blockchain network agree on the newly broadcasted block, which is essential for maintaining the blockchain's immutability and decentralization.

Theoretically, each node within a blockchain network retains a copy of the full ledger, ensuring uniformity across the system. The consensus algorithm establishes the necessary rules for validating new blocks and the transactions contained within them. This validation process ensures that each new block is accurate and consistent across all nodes within the blockchain network. The most important design goals for a consensus algorithm are achieving consistency among nodes, ensuring block authentication, and mitigating potential attacks against the algorithm's design. The common approaches to consensus algorithms include Proof of Work (PoW), Proof of Stake (PoS), Proof of Reputation (PoR), and other Byzantine Fault Tolerance algorithms.

2.4.1 Proof of Work

Proof of Work (PoW) was a consensus algorithm introduced with Bitcoin to address the Byzantine Generals' Problem (BGP), which involves a scenario where every node in a distributed system needs to reach consensus on a specific message. However, this consensus can be compromised by faulty or adversarial nodes that deliver incorrect or no messages, thereby threatening the consistency and safety of the distributed system. Bitcoin addresses the BGP by assuming that at least 51% of the network's computing power is controlled by honest nodes. Miners reach consensus by verifying the hash of the previous block and the current block, and checking if the PoW meets the required conditions. This process ensures that any new block shared in the consensus is valid. The approach solves the BGP with a cost of high computational complexity $O(2^n)$ [12]. PoW mechanism partially mitigates the DDOS attack because the attacker cannot predict which miner will successfully complete the PoW. Nevertheless, attackers could still potentially target the miners known to have significant

computational power.

An example of PoW is illustrated in Figure 2.4.1, which depicts a hash puzzle, also referred to as a *target* in PoW. The objective for miners is to compute a hash value that is less than or equal to the specified *target* value represented by the hash puzzle. A typical method to compute PoW involves three elements: the hash of the previous block, a nonce and the hash of the purposed block. The miner will add the three elements and inputs them into the SHA-256 hash function to compute a hash. The goal is to find a hash that is less than or equal to the *target* from last block. Specifically, the nonce is a 32-bit number selected by the miner, who increments it by one each time if the hash does not meet the requirement of *target* [13].

```
0000000000000fdb4a7dccff2faf6391478415ff73518a2e2bd3be60e1108e1d
```

Fig. 2.4.1: Target of PoW.

The effectiveness of the PoW approach is grounded in the uniform distribution of hash outputs generated by the hashing algorithms, which minimizes the probability of hash collisions. This uniform distribution also allows for control over the difficulty of the PoW task. Specifically, the difficulty can be adjusted by requiring the hash to have a certain number of leading zeros, which is defined by the *target*. The probability of a miner computing a hash that is lower than or equal to the target value in Figure 2.4.1 using a 32-bit nonce is $\frac{2^{32}}{2^{204}} = 2^{-172}$. It is the probability before the miner modifies the purposed block to satisfy the conditions of the *target*.

PoW addresses the BGP by imposing a heavy computational burden, utilizing properties of randomness and significant energy consumption. However, these characteristics make it less suitable for real-time systems where quick processing is essential.

2.4.2 Proof of Stake

Proof of Stake (PoS) is an alternative consensus algorithm designed to address the high computational costs associated with PoW. In recent implementations such as Ethereum, the leader for creating a new block is no longer determined by computational power but is instead chosen based on the amount of stake they hold. The

stake of a node consists of the number of digital tokens it holds. To determine the block leader in PoS, the Follow-the-Satoshi algorithm is adopted as the hash function, taking a seed as the input and outputting a token index. This index is then used to search the transaction history to find and select the current owner of the token as the leader [27]. One key difference between PoW and PoS is the penalty for dishonest behavior. In PoS, if an adversary attempts to create a faulty block, they risk losing their stake, as they must commit a certain amount of their tokens as a "bet" on the integrity and validity of the block they produce. Therefore, PoS resolves the BGP by discouraging faulty messages through a method of financial loss.

Nevertheless, blockchain with VANETs must apply this method cautiously because a traffic-related system with a reward-based blockchain could create perverse incentives. For instance, if adversaries wish to cause traffic congestion, they could simply buy tokens from a blockchain operating in a certain region. Subsequently, every CAV might hit the road to collect rewards, thereby facilitating the attack.

2.4.3 Proof of Reputation

Proof of Reputation (PoR) is a consensus algorithm that divides nodes into two types: regular nodes and consensus group nodes. A regular node can achieve a higher reputation by consistently demonstrating honest behavior. Once it joins the consensus group, it can verify and sign new blocks for consensus. It is better than a system that uses only reputation as a measurement. In paper [28], A trust-based system for VANETs is presented to identify and isolate malicious vehicles, allowing users to distinguish whether information from other CAVs can be trusted. To determine whether CAVs can be trusted, the authors present a system that categorizes trust levels into "Not Trust," "Trust," and "Middle Level." This trust level is determined by a fuzzy set system that outputs the trust level for each CAV.

A simple reputation-based system like this might not meet most of the VANET security characteristics, as it is vulnerable to repudiation attacks without a record of messages. Additionally, the integrity of the system is a concern since collaboration attacks on the trust levels can pose serious security risks.

PoR could mitigate the security problems posed by a solely reputation-based system by incorporating blockchain features. Aluko et al. [29] developed a reputation-based blockchain system by computing nodes' reputations through a blend of current peer ratings with historical data. This system allows reputation values to be fluid and reflective of recent behavior, while gradually lessening their impact over time. Moreover, the blockchain is public to everyone on the network to facilitate the transparency. A sidechain was used to store the values of reputation, eliminating the need for third-party storage. Another study, [30], introduced a cryptocurrency system that operates using PoR. The system employs a dual-blockchain design to distinguish between miners, who compute the hash, and group leaders, who create new blocks. A key block, created by the miner, is crucial for the election of new leaders within the group. Once the key block is created, the leader of the consensus group proceeds to create microblocks, which contain the cryptocurrency transactions. The reputation of the nodes is calculated based on their behavior over given time slots. Nodes can gain more reputation if they are honest or verified by a CA.

PoR is a feasible approach for VANET blockchain systems to achieve security characteristics. However, it needs to be carefully designed to mitigate all security issues. Table 2.4.1 compares three different consensus approaches.

Consensus Algorithm	Security	Efficiency
PoW	BGP is solved by property of hash	Very low as the computation of hash requires extensive computation power
PoS	BGP is solved by financial method	Fast, as the selection of leader does not require extensive computation
PoR	BGP is solved by repuation group	Relatively fast, still need to mitigate PoW for consensus, but transaction blocks' creation can be fast

Table 2.4.1: Comparsion between Different Consensus Algorithms.

2.5 Post Quantum Digital Signature Schemes

The digital signature scheme ECDSA is adopted as the digital signature method in Bitcoin, Ethereum, and many other blockchain systems. ECDSA was chosen because other digital signatures, such as RSA or the Digital Signature Algorithm (DSA), are not suitable for lightweight implementations. Specifically, ECDSA has a smaller key size compared to DSA and RSA, which enhances its performance since both RSA and DSA use the key as an exponent in their schemes [13]. However, with the emergence of quantum computing, ECDSA faces the potential of becoming outdated, as quantum computers with enough qubits could potentially crack the algorithm. This makes a MitM attack a feasible and powerful threat against all existing blockchains.

2.5.1 Shor's quantum algorithm

Shor's algorithm was designed to efficiently solve integer factorization and discrete logarithm problems, which are the underlying problems for RSA and ECDSA. The algorithm can solve these problems with a time complexity of approximately $O(n^3)$, which is significantly better than the classical algorithms' complexity of $O(e^{c \cdot \log^{1/3} n})$ [9]. The algorithm utilizes the superposition of all possible values of an unknown variable, with qubits representing multiple states simultaneously, as qubits remain in an indefinite state until observed. Quantum Fourier Transform (QFT) is used to exponentially speed up the process of finding periodicity in functions. A detailed implementation of Shor's algorithm can be found in [9].

2.5.2 CRYSTALS-Dilithium

Dilithium is one of the digital signature schemes submitted to NIST for the post-quantum digital signature standard. It is based on the computational hardness of finding short vectors in lattices. The algorithm was designed base on the quantum algorithms that requires virtually as much space as time, which is unrealistic with current advancement in quantum computer. As for the security of dilithium, it can be demonstrated that within the (classical) random oracle model, Dilithium achieves

SUF-CMA security, relying on the difficulty of the standard MLWE and MSIS lattice problems [14].

Although there is no officially recognized post-quantum digital signature scheme yet, we can still use the implementation of Dilithium to test its suitability as a digital signature scheme within blockchain applications in VANET.

2.6 Summary

In this chapter, we introduced two types of traffic data collection systems: on-road sensor and FCD. We also provided background knowledge on VANET, including its communication model, key components, and wireless standards. Then, we introduced current research fields in blockchain applications in VANET. Most research focuses on non-real-time approaches or uses customized consensus algorithms to preserve the security features provided by blockchain while reducing the burden of PoW. After introducing blockchain applications, we explored various popular consensus algorithms used in major blockchain systems, including PoW, PoS, and PoR, each with its own advantages and disadvantages in terms of security and efficiency. We also discussed the potential obsolescence of ECDSA with the emergence of quantum technology. Lastly, we introduced Dilithium as the digital signature algorithm we plan to use for our experiments. In the next chapter, we will introduce the proposed approach for applying blockchain systems to VANET, focusing on its design architecture

CHAPTER 3

Proof-of-Presence: A Blockchain Based Traffic Data Collection System

To develop a traffic data collection system within VANET, the immediacy of the collected data and the system's security are major concerns. Employing blockchain can enhance certain security properties; however, traditional consensus approaches may introduce significant overhead, compromising the real-time requirements of the system. To balance the need for real-time data collection with the security benefits of consensus algorithms, a customized consensus algorithm is essential to minimize overhead. Furthermore, ensuring the authenticity and integrity of the data sources is crucial to complement the adaptations made in the customized consensus algorithm. This thesis aims to provide a balanced design approach that addresses the real-time requirements of a traffic data collection system and preserves the security features that blockchain technology offers to VANET. Here are the goals of the thesis:

- **Providing** the architecture of a blockchain-based traffic data collection system in VANET.
- **Providing** different message protocols to ensure security.
- **Designing** a customized consensus algorithm.
- **Explaining** the ideas behind collecting reliable source of messages.

- **Analyzing** the security and performance of the purposed blockchain system.

3.1 VANET Elements

In VANET architecture, CAVs and RSUs play pivotal roles in the collection of traffic data. This data primarily derived from BSMs, which encapsulate critical information such as timestamps, GPS coordinates, speed and safety-related data. The choice to utilize DSRC for transmitting the BSMs is strategic, owing its effectiveness in supporting real-time vehicular communication. Figure 3.1.1 illustrates BSMs' delivery by V2V and V2I. CAVs deliver BSMs to RSUs and other CAVs to ensure mutual safety.

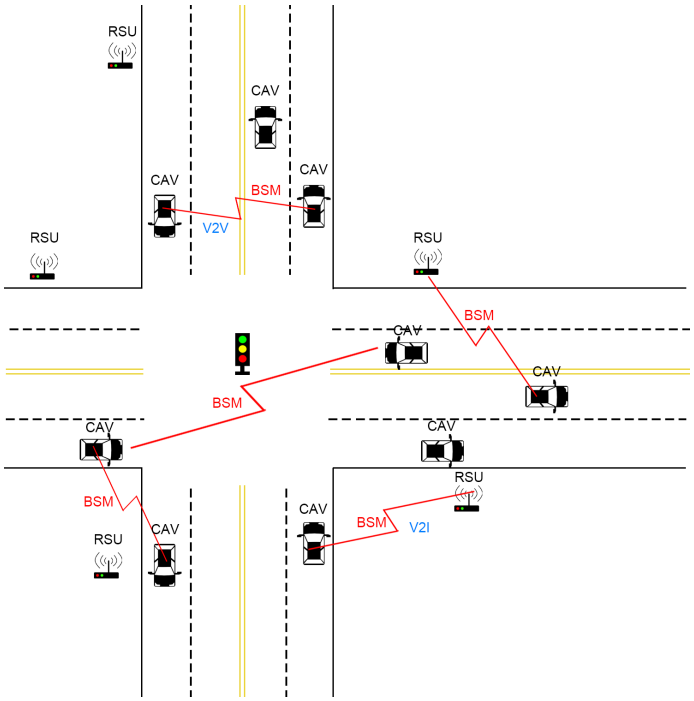

Fig. 3.1.1: Delivery of BSMs in VANET.

Furthermore, the protocol stack for DSRC, along with C-V2X communications, encompasses both TCP and UDP. The dual support facilitates the development and deployment of customized protocol messages tailored for specific needs within the VANET.

In our proposed system, BSMs are first delivered to relevant entities, after which

CAVs and RSUs process these messages to complete their specific tasks within the blockchain and VANET. The roles of CAVs and RSUs in our system, from the perspective of VANET, will be discussed in the next few subsections.

3.1.1 Role of CAVs

CAVs play a central role in collecting traffic data because they are an integral part of the traffic. BSMs are generated by CAVs every 100 milliseconds, resulting in a data volume that is too large to process continuously. To manage this, BSMs are gathered at fixed intervals, and only the BSMs from specific moments are processed in the blockchain to reduce system overhead. Figure 3.1.2 illustrates an example of the BSM collection intervals, with BSMs being collected every three timeslots.

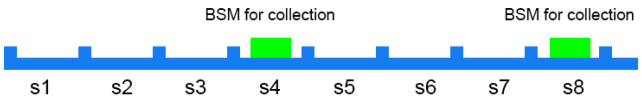

Fig. 3.1.2: BSM Collection Interval.

In our system, once the CAVs receive BSMs produced at a specific moment for collection, they begin verifying the BSMs and send the legitimate ones to the group members in the VANET, who are responsible for controlling the blockchain. Additionally, the CAVs send their sensor data, including camera and radar data, to the group members for cross-validation of the authenticity of the BSMs.

3.1.2 Role of RSUs

RSUs play an important role in the traffic data collection system. Although they may have fewer sensors than CAVs due to their primary role in communication within VANETs, they are more trustworthy and controllable because they are managed by a CA and are part of the stationary infrastructure. The role of RSUs in our system is to complement the gathering of BSMs from CAVs and to verify them. They also

function as enhanced processing units when CAVs are underpowered for verifying certain types of messages.

3.2 Architecture of Proof-of-Presence Blockchain System

Proof-of-Presence (PoPr) is built on the foundation of PoR. However, PoR cannot be directly applied to the traffic data collection system because its reputation evaluation rules are not suitable. A direct approach, such as verifying BSM signatures to assign reputation, does not adequately reflect the honesty of CAVs or RSUs in a fully decentralized blockchain. Additionally, involving a CA too heavily in the blockchain system, as seen in most VANET approaches, raises concerns about insufficient decentralization. This could compromise the privacy and anonymity of nodes, thereby undermining the fundamental purpose of blockchain.

PoPr offers a solution to mitigate all the aforementioned problems. The outline of the PoPr blockchain is provided below:

- Joining the blockchain network
- BSM and BSM verification messages
- Time chain
- BSM data chain
- PoPr consensus algorithm

The overview of the PoPr BSM data blockchain Consensus is depicted in Figure 3.2.1. Within this system, BSMs are verified and subsequently integrated into blocks. This process facilitates the evaluation of the reputation of every node in the system.

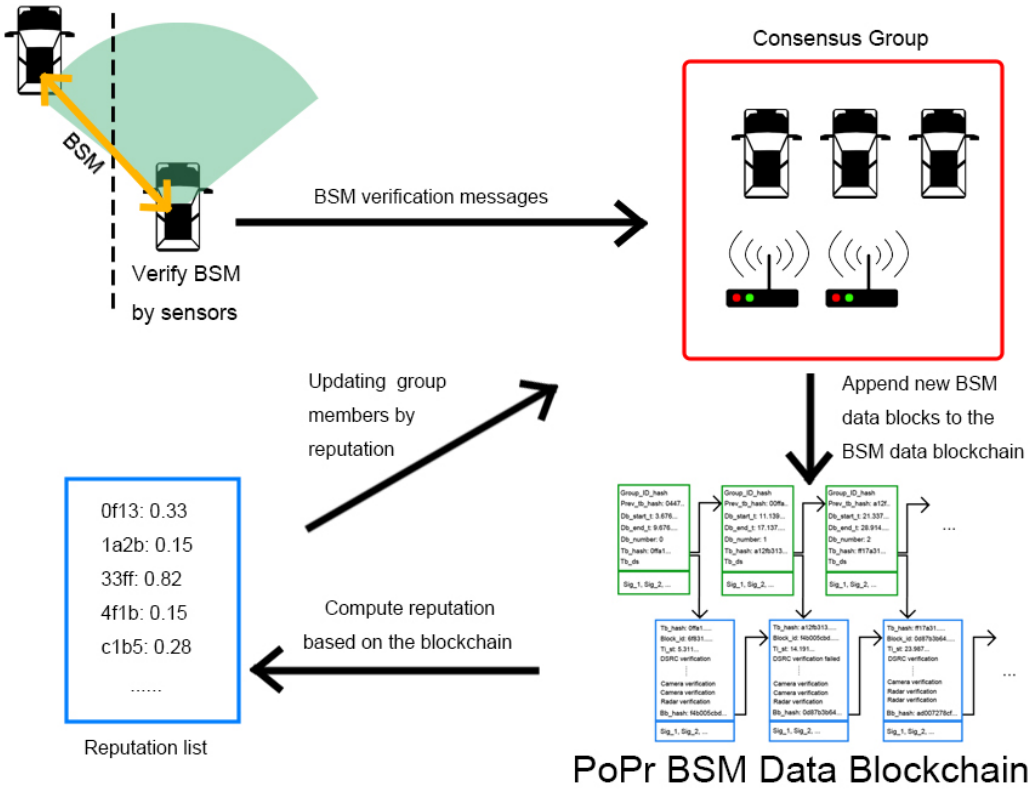

Fig. 3.2.1: Overview of PoPr BSM Data Blockchain Consensus.

3.2.1 Joining the Blockchain Network

In the proposed blockchain system, security is not primarily derived from the verification process associated with joining the network. Instead, it mirrors the approach used in cryptocurrency blockchain systems. The key difference lies in the actions of newly joined nodes: rather than broadcasting transactions, these nodes broadcast their BSMs that include their public key. The format and purpose of different messages will be further introduced in the following subsection.

3.2.2 BSM and BSM verification messages

The basic safety message (BSM) in our system is presented as follow:

$$\{PK\_ID|Orientation|Timestamp|BSM\_GPS|BSM\_hash|BSM\_sig\}$$

The *PK_ID* represents the public key generated by a node, serving also as the node's identifier within the network. *Orientation* indicates the current orientation of the CAV, providing crucial data to ensure the safety of other CAVs and entities within the VANET. The *Timestamp* records the moment when the blockchain system schedules the collection of BSMs from the VANET, reflecting a structured interval-based data gathering approach. *BSM_GPS* represents GPS coordinates of the CAV sender, including both latitude and longitude. This essential data is crucial for processing various tasks within the blockchain and VANET. *BSM_hash* are the hash value computed from (*Orientation*, *Timestamp*, *BSM_GPS*), and *BSM_sig* is the digital signature of the *BSM_hash*. Hashing the essential values in BSM ensures that any tampering with the values is reflected in the hash, and signing the hash itself secures the integrity of the data, as any tampering with the hash will invalidate both the signature and values contained in the BSMs.

Given the design of BSM in our system, CAVs can retrieve critical data from these messages for their decision-making processes. Additionally, the inclusion of the public key in *PK_ID* enables CAVs to verify the authenticity of each BSM. The design not only ensures the integrity of the messages but also provides a traceable method for confirming their authenticity.

Conversely, BSMs must be verified by other CAVs or RSUs to confirm the sender's actual "presence" on the road. This verification is crucial to reduce the likelihood of the network being overwhelmed by fake BSMs In our system, we introduced three distinct types of BSM verification messages for verifying the BSMs. These includes: DSRC verification messages, Camera-based verification message and Radar-based verification message.

3.2.2.1 DSRC Verification Message

The DSRC verification message is collected using DSRC in VANET, the message's format is presented below:

$$\{PK\_ID|Ori\_BSM|Veri\_hash|Veri\_sig\}$$

The *PK_ID* denotes the public key and identifier of the verifier, which includes both CAVs and RSUs within the network. *Ori_BSM* denotes the original BSM received from the CAVs in the range of DSRC. Figure 3.2.2 illustrates how BSMs is delivered through DSRC. The *Veri_hash* is the hash computed from (*Orientation*, *Timestamp*, *GPS*, *BSM_sig*). Subsequently, *Veri_sig* denotes the digital signature of the *Veri_hash*.

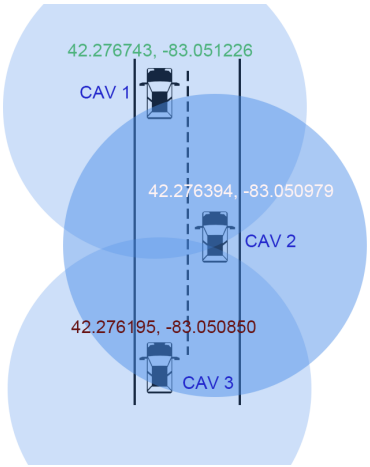

Fig. 3.2.2: Delivery of BSM via DSRC.

The design of the DSRC verification message is straightforward: the verifier checks the signature of the BSM received, and then declares that they have received a BSM from a CAV in the blockchain network by sending this verification message to consensus group members. This method doe not allow the verifier to pinpoint the exact location from which the CAV transmitted the BSM, but only able to confirm the BSM was sent within the range of the DSRC. Consenquently, the "presence" evidence it provides is considered weak. However, this DSRC verification message serves as the foundation for the subsequent two types of verification messages, as the verification process is structured hierarchically.

3.2.2.2 Camera Verification Message

The camera verification message is collected using both front and rear camera sensors of CAVs, as well as front cameras of RSUs. The message's format is denoted below:

$$\{PK\_ID|Ori\_BSM|Veri\_GPS|Cam\_hash|BSM\_azi|Veri\_azi|Veri\_hash|Veri\_sig\}$$

Similar to the DSRC verification message, *PK_ID* is the public key and identifier of the verifier. Additionally, *Ori_BSM* denotes the original BSM received from the nearby CAVs. Conversely, *Cam_hash* includes both hash of the images from the front camera and rear camera. *Veri_GPS* represents the GPS coordinates of the verifier, while *BSM_azimuth* denotes the azimuth computed from the verifier's GPS coordinates to the GPS coordinates of the BSM sender. Moreover, *Veri_azi* represents the azimuth from the output of machine learning models, which use images captured by camera sensors as input. Ultimately, *Veri_hash* is the hash computed from (*Cam_hash*, *Veri_GPS*, *Veri_azi*, *BSM_hash*). Following the hash computation, *Veri_sig* is the signature of the *Veri_hash*.

The azimuth, derived from both the GPS data and verifier's camera data, is essential for verifying the BSM in camera verification message. The azimuth computed from BSM is derived using haversine formula, which computes both the distance and azimuth on earth. Algorithm 3.2.1 details the method for computing azimuth using the latitude and longitude of two GPS coordinates.

Algorithm 3.2.1 Azimuth Computation

```
1: function AZIMUTH_COMPUTATION(lat1, lon1, lat2, lon2)
2:     convert latitude and longitude to radians:
3:         lat1, lon1, lat2, lon2 = map(radians, [lat1, lon1, lat2, lon2])
4:         dlon = lon2 - lon1
5:         x = cos(lat2) * sin(dlon)
6:         y = cos(lat1) * sin(lat2) - sin(lat1) * cos(lat2) * cos(dlon)
7:     initial_bearing = atan2(x, y)
8:     convert to compass bearing (0 to 360 degrees):
9:         compass_bearing = (degrees(initial_bearing) + 360) % 360
10:    return compass_bearing
11: end function
```

Once the verifier computes the azimuth from GPS coordinates, it then compares this result with the azimuth computed from the camera sensor. The azimuth result from camera is derived from two machine learning models. Figure 3.2.3 illustrates the procedure employed by machine learning models to determine the azimuth from camera images. Initially, the camera image undergoes analysis by an object detection algorithm, such as YOLO, which identifies the CAVs within the image. Upon receiving the output from the object detection model, the coordinates of the bounding box are then used as input to another model specifically trained to compute the azimuth by these coordinates.

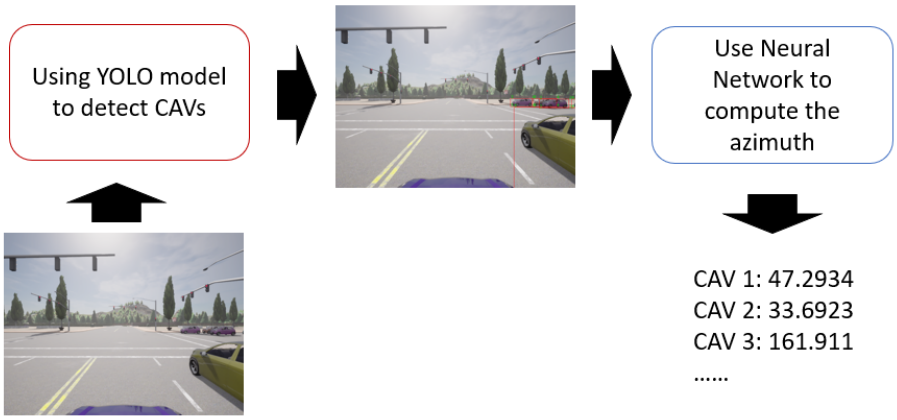

Fig. 3.2.3: Machine Learning Models for Determining Azimuth from Camera Images.

Given the fact that azimuth results from machine learning models may not precisely match those derived from GPS coordinates, the verifiers incorporate a degree of tolerance when comparing azimuth values. If the azimuth calculated from image data falls within this tolerance range, it is concluded that the CAV which sent the BSM has been accurately identified by the image data. Consequently, the verifier can then generate a camera verification message to be sent to consensus group members.

3.2.2.3 Radar Verification Message

The radar verification message is collected using both front and rear radar sensors of CAVs, as well as front radars of RSUs. The format of this message is detailed below:

$$\{PK\_ID|Ori\_BSM|Veri\_GPS|Radar\_pts|BSM\_azi|BSM\_dep|Veri\_hash|Veri\_sig\}$$

The definitions of *PK_ID*, *Ori_BSM*, *Veri_GPS* and *BSM_azi* remain consistent with those outlined in the camera verification message. Moreover, *Radar_pts* refers to the radar points detected by the verifier's radar, which include the speed, depth and azimuth of detected objects. In our system, both depth and azimuth measurements obtained via radars are employed to verify the sender of the BSM. The *BSM_azi* and *BSM_dep* represent the azimuth and depth, respectively, computed from the verifier's coordinates to the sender's coordinates. Ultimately, *Veri_hash* is the hash computed from (*Radar_pts*, *Veri_GPS*, *BSM_azi*, *BSM_dep*, *BSM_hash*). The *Veri_sig* is the signature of the *Veri_hash*.

To verify the BSM in radar verification, the verifier searches for radar points captured at the moment the BSM was received. Similar to camera verification, a degree of tolerance is applied when comparing the azimuth and depth of radar points with those derived from coordinates. Once a match is found for both azimuth and depth, the verifier generates the radar verification messages and sends them to the consensus group members. Figure 3.2.4 illustrate the process of verifying a CAV by Radar.

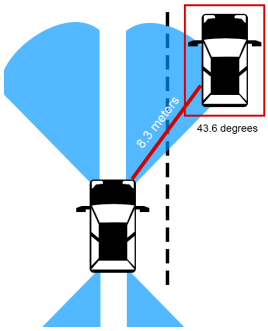

Fig. 3.2.4: Radar Verification Process.

3.2.2.4 Hierarchy and Pair Mechanisms of Verification Messages

A simple hierarchical system enhances the security of verification messages. Initially, DSRC verification messages serve as the foundational layer for both camera and radar verification messages. This implies that if a DSRC verification message exists for a specific BSM, then a camera verfication message or a radar verification message may also be generated for the same BSM. This design eliminates the possiblity for an adversary to generate indefinite camera verification messages or radar verification messages without a BSM that can be detected by other CAVs or RSUs. Figure 3.2.5 illustrates the simple hierarchy of verification messages

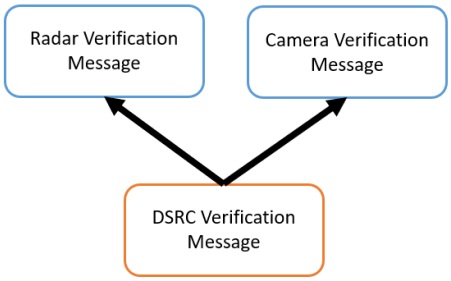

Fig. 3.2.5: Structure of Messages Verification Hierarchy.

Another mechanism that ensures the correctness of DSRC verification messages is termed "pair verification". Given the nature of the BSMs transmitted via DSRC, both the sender and verifer—assuming they are CAVs—should receive each other's BSMs. Therefore, DSRC verification messages delivered to consensus group members must be paired. Consequently, any DSRC verification message that failed to pair with at least one other BSM will be discarded. An example of pair mechanism is illustrated in Figure 3.2.6.

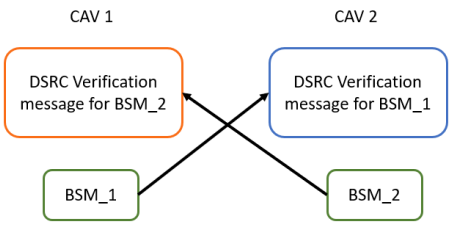

Fig. 3.2.6: Pair Mechanism.

In the example, *CAV_1* sends *BSM_1* to *CAV_2*, and *CAV_2* sends *BSM_2* to *CAV_1*. Subequently, *CAV_1* creates a DSRC verifcation message for *BSM_2*, and *CAV_2* creates a DSRC verification message for *BSM_1*. Thus, the DSRC verification message between *CAV_1* and *CAV_2* is paired and deemed legitimate.

The pair mechanism ensures that an adversary must generate DSRC verification messages in pairs, eliminating the possibility of creating an indefinite number of unrelated DSRC verification messages.

3.2.3 Time Chain

There are two blockchains utilized in the purposed system. The *Time Chain* is one such blockchain, featuring several crucial elements designed to ensure the accuracy of data blocks, the real-time performance of the blockchain, and the correctness of leader election processes. It is a substitution of PoW to ensure the correctness of consensus. The time block in the time chain are created by group members during the time block consensus. The structure of the time block in the time chain are illustrated in Figure 3.2.7.

Group_ID_hash | Prev_tb_hash | Db_start_t | Db_end_t | Db_number | Tb_hash | Tb_ds | Group_id || Tb_approve_sig

Fig. 3.2.7: Time Block Structure.

The *Group_ID_hash* is the hash of consensus group members' public keys, arranged in order of reputation from highest to lowest. This ensures that the time block is created by the group members designated to initiate the time block consensus, based on their reputation. *Prev_tb_hash* is the hash of the previous time block. For the genesis block within the time chain, the hash of "0" is used as the *Prev_tb_hash*. The *Db_start_t* represents the agreed-upon start time by consensus group members for initiating the creation of the next BSM data block. This start time must occur after the completion of the time block consensus. Conversely, the *Db_end_t* is the max time for the creation of BSM data block. The interval between *Db_start_t* and *Db_end_t* is determined based on the current number of

nodes and network status, which is assessed by factors such as network quality, including response times between nodes and the average processing time across VANET. Moreover, *Db_number* is the new BSM data block number associated with this time block. *Group_id* servers as both the public key and identifier of this group member who purposed this time block. *Tb_hash* is the hash value of (*Group_ID_hash*, *Prev_tb_hash*, *Db_start_t*, *Db_end_t*, *Db_number*, *Group_id*), and *Tb_ds* is the digital signature of the *Tb_hash*. *Tb_approve_sig* consists of the approved signatures from consensus group members, excluding the block purposer. Figure 3.2.8 depicts an example of a time chain. In this structure, the *Prev_tb_hash* links each time block to its predecessor by using the previous *Tb_hash*. This chaining ensures the continuity and integrity of the time chain over time.

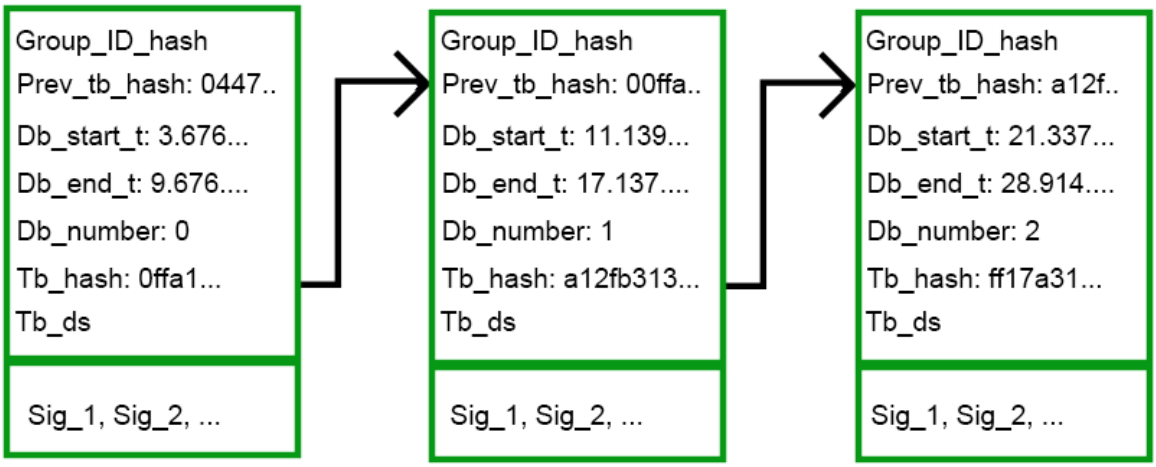

Fig. 3.2.8: An Example of Time chain.

3.2.4 BSM Data Chain

Another type of blockchain utilized in our system is the *BSM Data Chain*. This chain records the entries of verification messages for BSMs that are collected and verified by consensus group members, It serves as a crucial source of traffic data collection and reputation accumulation. The structure of the BSM data block illustrated in Figure 3.2.9.

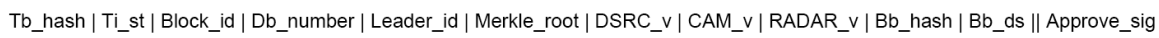

Fig. 3.2.9: BSM Data Block Structure.

In our purposed design, *Tb_hash* represents the hash of the time block to which each BSM data block is chained. Each BSM data block is created within the interval defined by *Db_start_t* and *Db_end_t* of the corresponding time block. The *Ti_st* represents the timestamp when this block is first purposed by the leader of the consensus group members. *Block_id* is the hash of the previous BSM data block. For the genesis block within the BSM data block chain, hash of "00" is used as the genesis *Block_id*. *Db_number* is the BSM data block number assigned by the associated time block. *Leader_id* is the public key and identifier of the leader who purposed this BSM data block. *Merkle_root* is the root hash of the merkle tree structure that contains the hash of all verification messages.

In this merkle tree, each leaf node represents the hash of verification messages within the BSM data block, each parent node in the tree holds the hash derived from combining the hashes of at most two child nodes. The hashing process continues upwards through hierarchy of nodes until the *Merkle_root*. Consequently, if the adversary attempts to alter the block by introducing faulty or fake verification messages, such alternations will change the hashes of the affected leaf nodes. This modification propagates up the tree, altering the hashes of the parent nodes, and eventually affect the *Merkle_root*. Therefore, the new merkle root will not match with the original *Merkle_root* and indicating the block entries are tampered. Figure 3.2.10 illustrates the merkle tree structure.

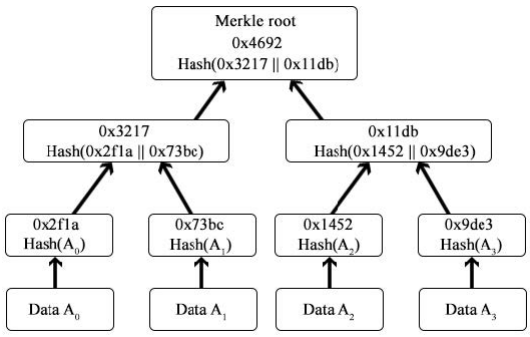

Fig. 3.2.10: Merkle Tree Structure [13].

The next three parts *DSRC_v*, *CAM_v* and *RADAR_v* are the three types of verification messages for the BSMs received at the specific moment. The types of the verification messages are introduced in the aforementioned sections. Another type of message, referred to as *BSM's faulty message*, is introduced in the section discussing the PoPr Consensus Algorithm. *Bb_hash* is the hash value of (*Tb_hash*, *Ti_st*, *Block_id*, *Db_number*, *Leader_id*, *Merkle_root*). The *Bb_ds* is the digital signature of the hash *Bb_hash*. Finally, *Approve_sig* consist of the approved signatures from consensus group members, excluding the leader.

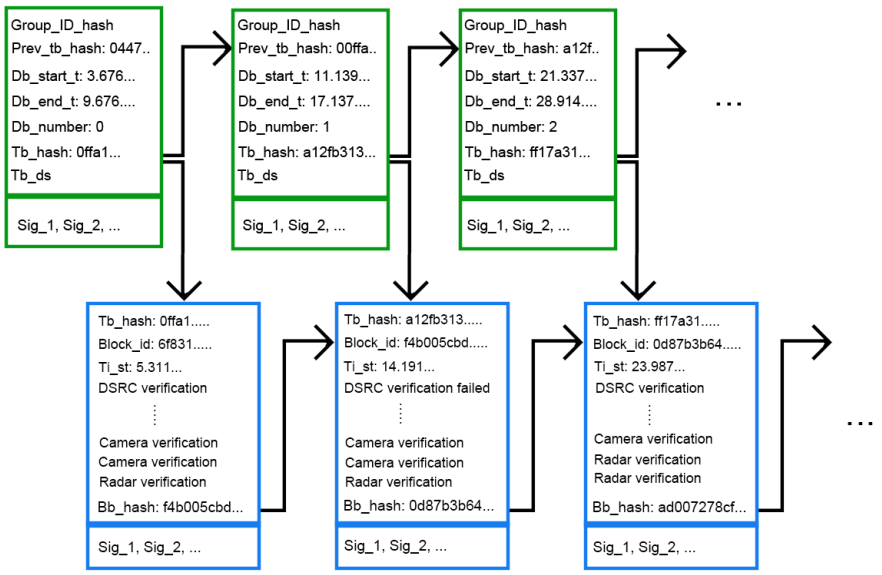

Fig. 3.2.11: An Example of BSM Data Chain.

Figure 3.2.11 illustrates an example of BSM data chain, where each BSM data

block is linked not only to the previous BSM data block but also to a specific time block. The BSM data block is created at *Ti_st*, which falls within the interval defined by *Db_start_t* and *Db_end_t* of the associated time block.

3.2.5 PoPr Consensus Algorithm

PoPr consensus algorithm is the core function of the PoPr blockchain system and comprises four key procedures: reputation assessment and consensus group members assignment, time block consensus, BSM block leader selection, and BSM data block consensus. The sequence of procedures are given below:

1. Reputation assessment and consensus group members assignment
2. Time block consensus
3. BSM data block leader selection
4. BSM data block consensus

3.2.5.1 Reputation Assessment and Consensus Group Members Assignment

Reputation is a crucial metric in the PoPr consensus algorithm, serving to measure the honesty of each node within the VANET. Nodes with higher reputations are considered trustworthy; their BSMs are utilized for traffic data collection, and they may also participate as group members in the decision-making processes for consensus and creation of time blocks and BSM data blocks. Conversely, nodes with lower reputation are identified as faulty or hostile and subsequently excluded from traffic data collection activities. To accurately assess the reputation of nodes, it is necessary to calculate this metric based on their BSM's verification messages and *BSM's faulty messages*.

BSM Faulty Message represent a specific category, distinct from the three types of BSM verification messages. These occur when CAVs or RSUs are unable to verify a BSM claimed to be received by other nodes. Figure 3.2.12 illustrates such a BSM

faulty scenario: CAV 1 report having received a BSM from CAV 2 and sends a DSRC verification message to confirm it. In contrast, CAV 3, which disputes receiving this BSM, does not send a verification message.

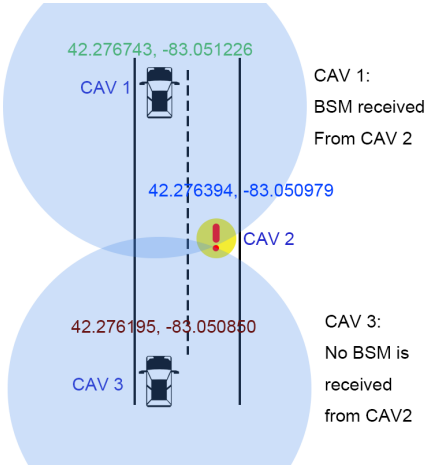

Fig. 3.2.12: A DSRC Faulty Scenario.

BSM faulty messages are evaluated and recorded by the consensus group members within the network. When a BSM faulty message is created, it can lead to a reduction in the reputation of the specific node that sent the BSM.

The total reputation computed from BSM data blockchain for a node is given in follow equation:

$$RS = \sum_{i=max\_t}^{n} \sigma(\sum_{j=1}^{m}(d_{i,j}\cdot w_d + c_{i,j}\cdot w_c + r_{i,j}\cdot w_r + ft_{i,j}\cdot w_f) + (D_{i,j}\cdot v_d + C_{i,j}\cdot v_c + R_{i,j}\cdot v_r))e^{-(n-i)^2}$$

RS represents the total reputation computed from the BSM data blocks from max_t to n. Here, max_t denotes the oldest BSM data block used in the reputation calculation, emphasizing the importance of basing reputation on the most recent blocks to ensure relevance. σ is the sigmoid function, the function is given below:

$$\sigma(x) = \frac{1}{1 + e^{b\cdot(a-x)}}$$

where a and b are coefficient factors. The variables j to m represents every entry in a BSM data block. The variables $d_{i,\,j}$, $c_{i,\,j}$ and $r_{i,\,j}$ denote whether the entry i,

j is a DSRC, camera or radar verification message for verifying the node's BSM, respectively. Conversely, $D_{i,\,j}$, $C_{i,\,j}$ and $R_{i,\,j}$ denote where the entry i, j is a DSRC, camera or radar verification message sent from the node's BSM, respectively. Each variable is set to 1 if the entry corresponds to the specified message type, and 0 otherwise. The variables w_d, w_c and w_r indicate the reputation weight assigned to DSRC, camera or radar verification messages for the BSM sender, respectively. The variables v_d, v_c and v_r denote the reputation weight assigned to DSRC, camera or radar verification messages for the BSM verifier, respectively. The variable $ft_{i,\,j}$ is whether an entry is a faulty message. The corresponding reputation weight for such faulty messages is denoted by w_f. The term $e^{-(n-i)^2}$ denotes an exponential decay function, where the exponent is squared to emphasize the impact of more recent BSM data blocks on the node's reputation.

The reputation weights are adjusted based on the number of nodes joining the blockchain network. This adjustment accounts for the increased volume of DSRC verification messages contributed by more CAVs or RSUs. However, since DSRC verification messages are the weakest in verifying BSMs, their reputation weight should not be significantly increased, even though they are easier to verify. This is reflected in the following equations:

$$v_d = \frac{v_{d_{ori}}}{\epsilon}, v_c = \frac{\epsilon}{2v_{c_{ori}}}, v_r = \frac{\epsilon}{v_{r_{ori}}}$$

where $v_{d_{ori}}$, $v_{c_{ori}}$, $v_{r_{ori}}$ denote the original reputation weight assigned to DSRC, camera or radar verification messages for the BSM verifier, respectively. ϵ is the current amount of nodes joining the PoPr blockchain network. The equations reflects how the reputation weights for DSRC, camera, and radar verification messages adapt to changes in network size. When fewer nodes are active in the network, the reputation weights for DSRC, camera, and radar verifications are similar. However, as more nodes join the network, the reputation weight assigned to DSRC verification decreases due to its lower verification strength, while the weights for camera and radar verifications increase, reflecting their greater reliability in a larger network.

Moreover, more reputation weight should be assigned to the BSM sender, as the "presence" of a CAV is crucial in ensuring that the BSM originates from a legitimate source.

The full computation algorithm of reputation is given in the Algorithm 3.2.2. The notations are given in table 3.2.1

Notation	**Description**
$node$	the node for compute the reputation
CA_{ext}	the external source of trust, mostly for RSUs
(a, b)	coefficient factors for sigmoid functions
u	reputation factor for weights of BSM sender
ϵ	the amount of nodes joining the network
$v_{d_{ori}}, v_{c_{ori}}, v_{r_{ori}}$	the original reputation weights for DSRC, camera, radar verification messages of BSM verifier
$w_{d_{ori}}, w_{c_{ori}}, w_{r_{ori}}$	the original reputation weights for DSRC, camera, radar verification messages of BSM sender
w_f	the reputation weight for faulty message
max_t	the oldest block used to compute the reputation
BSM_c	the BSM data chain
$Time_c$	the time chain

Table 3.2.1: The Notations for Reputation Algorithm.

Algorithm 3.2.2 Reputation Computation Algorithm

Input: $node, CA_{ext}, a, b, u, \epsilon, v_{d_{ori}}, v_{c_{ori}}, v_{r_{ori}}, w_{d_{ori}}, w_{c_{ori}}, w_{r_{ori}}, w_f, max\_t, BSM_c$ and $Time_c$

Output: Reputation $R \in [0, 1]$ for a certain node

1: $d, c, r, ft \leftarrow BSM_c(max\_t, node)$
2: $Ml \leftarrow BSM_c(max\_t, node), Time_c(max\_t, node)$
3: $\sigma(x) = \frac{1}{1+e^{b\cdot(a-x)}}$
4: $v_d = \frac{v_{d_{ori}}}{\epsilon}, v_c = \frac{\epsilon}{2v_{c_{ori}}}, v_r = \frac{\epsilon}{v_{r_{ori}}}$
5: $w_d = \frac{u\cdot w_{d_{ori}}}{\epsilon}, w_c = \frac{u\cdot\epsilon}{2w_{c_{ori}}}, w_r = \frac{u\cdot\epsilon}{w_{r_{ori}}}$
6: $R_{BSM} = \sum_{i=max\_t}^{n} \sigma(\sum_{j=1}^{m}(d_{i,j}\cdot w_d + c_{i,j}\cdot w_c + r_{i,j}\cdot w_r + ft_{i,j}\cdot w_f) + (D_{i,j}\cdot v_d + C_{i,j}\cdot v_c + R_{i,j}\cdot v_r))e^{-(n-i)^2}$
7: $R_{Time} = \sum_{i=max\_t}^{n} Ml_i \cdot w_l$
8: $R = min(1, max((R_{BSM} + R_{Time}) + CA_{ext}, 0))$

In the reputation compuation algorithm, $CA_{ext} \in [0, 1]$ represents the external source of reputation, such as when RSUs are verified by a specific CA and consequently gain reputation based on this verification. The sigmoid function are used to output the reputation because we aim to normalize the reputation value to $[0, 1]$, making the value of CA_{ext} easier to adjust. *Ml* represents the number of times a node has failed to create a block when it is the leader. The weight w_f is the reputation deduced when failed to create a block when the node is a leader. R_{time} is the reputation weight computed by aggregating the number of time blocks that remain unlinked with a BSM data block when the node is the leader.

The reputation algorithm is executed immediately after the creation of BSM data block because the consensus group members are selected based on their reputation. The method used to determined the consensus group members are straightforward:

1. Decide the group size G by the size of the blockchain network
2. Sort the reputation from highest to lowest
3. Select the first G nodes as the consensus group members

In the case where two or more nodes have same reputation, each nodes will compute the hash of the public key *pk* of the nodes with same reputation, the lastest time

block's *Db_start_t* and the nodes' reputation *R*. The node with the larger hash value will be sorted ahead of the node with the lower hash. The equation for the hash is given below:

$$H_{order} = hash(pk + Db\_start\_t + R)$$

In this way, nodes with the highest reputation become the consensus group members. The results of group member selection should be consistent across all nodes because the BSM data blocks are consistent within the blockchain network.

3.2.5.2 Time Block Consensus

After the consensus group members are assigned, the group will begin the consensus process for the time block. Each group member will send a proposed time block with a different *Db_start_t* to every other consensus group members. The consensus group members will wait a time interval t_{tb}. The timeout interval is decided with the equation below:

$$t_{tb} = base\_t\_vote + prev\_tb\_hash \mod max\_t\_vote\_mod$$

Where *base_t_vote* is the base time interval required for a time block consensus to complete, and *max_t_vote_mod* is the maximum time allowed for a random timeout, which is determined based on the *prev_tb_hash*. They will then vote on whether they received every consensus group member's purposed time block, or at least one purposed time block from another group member. If both conditions are fail to match, the consensus group members will initiate another time block consensus with t_{tb} interval, continuing until timeout occurs again.

Moreover, if the two or more purposed time block have the same *Db_start_t*, each consensus group member will compute the hash by combining *node_pk* and *Db_start_t*, where *node_pk* is the public key of the consensus nodes. The consensus group member will then select the larger hash as the new time block.

The consensus group members vote on the time block by sending an approval message to the group member whose new time block they accept. When a consensus

group member receives $\lceil \frac{2\cdot(G-1)}{3} \rceil$ votes, they will send the approval message to all consensus group members to stop their timeout and proceed to the next step. The time block consensus algorithm is given in Algorithm 3.2.3

Algorithm 3.2.3 Time Block Consensus Algorithm

Input: Consensus group members list G_{list}

1: Purposed time block TB_{new} is created
2: Send purposed time block TB_{new} to G_{list}
3: Wait to receive purposed time block and put them in TB_{list}
4: **if** $TB_{list} == G_{list}$ or $TB_{list} > 2$ when timeout **then**
5: Accept the purposed time block with smallest Db_start_t
6: **if** same Db_start_t for at least two TB_{new} **then**
7: Compute the $hash(node\_pk + Db\_start\_t)$ for same Db_start_t
8: Compare the hash, the largest one will be the new time block
9: **end if**
10: Send approval message to the accepted time block's purposer
11: **else**
12: Start a timeout again by set the timeout time to $curr\_time + t_{tb}$
13: **end if**
14: **if** the purposed TB_{new} receives $\lceil \frac{2\cdot(G-1)}{3} \rceil$ approval messages **then**
15: Boardcast the $\lceil \frac{2\cdot(G-1)}{3} \rceil$ approval messages received to G_{list}
16: **end if**

3.2.5.3 BSM Data Block Leader Selection

The consensus group leader, who creates the new BSM data block, is selected from among the group members using the following equation:

$$G_{leader} = hash(Bb\_hash + Db\_number + Db\_start\_t) \mod G$$

Where Bb_hash is the latest hash of BSM block, Db_number is the block number of the latest BSM data block, and Db_start_t is is the start time of creation for the latest BSM data block in the latest time block. After acquiring G_{leader}, which is the index of the consensus group member, the consensus group member will instantly know who the next leader is and will start the timeout process for the creation of BSM data block.

The hash includes the latest BSM data block to ensure randomness. Since the BSMs and BSM verification message contains high entropy when the CAVs are distributed throughout the city, the randomness of their route planning enhances the randomness in the *Bb_hash*. Moreover, to prevent passive attacks from a leader who intentionally avoids creating a new BSM data block, *Db_number* and *Db_start_t* are used as additional parts of the hash input. These variables are continuously updated if a BSM data block timeout occurs. Specifically, *Db_start_t* also introduces a degree of randomness, as the quality of VANET is not entirely controllable in a real-world setting.

3.2.5.4 BSM Data Block Consensus

To achieve a BSM data block consensus, the leader of the consensus group and other group members collect the BSM verification messages sent from nodes in the blockchain. Once the *Db_start_t* is reached, the leader and group members will cease collecting BSM verification messages and begin generating BSM faulty messages based on the received BSM verification messages. When leader is ready, they present the new BSM data block to other group members for voting. Similar to time block consensus, once the leader receives $\lceil \frac{2 \cdot (G-1)}{3} \rceil$ votes. They will immediately boardcast approval messages to other group members to halt their timeout. Additionally, the leader will boardcast new BSM data block is ready to other non-group members for collecting the new BSM data block.

The voting conditions for the new BSM data block is presented as follow:

- Verify that the block was received during the interval between *Db_start_t* and *Db_end_t* in the linked time block.
- At least 90% of the entries of messages in the new BSM data block are the same as those in the current group members' entries. This limit can be changed by group members' consensus.
- The entries in the blocks must match with the pair and hierarchy mechanism.

- The merkle root of the block must correspond to the entries.
- Verify the hash and digital signature of the block to ensure their validity.

Given the voting conditions, the consensus group will evaluate the BSM data block they have received according to these conditions. If the conditions are not met, the consensus group members will not issue an approval message, and the leader will need to submit a revised BSM data block if they wish to receive approval. However, if the time specified by Db_end_t in the linked time block is reached, a new consensus process for the time chain will start. If the leader fails to create a BSM data block linked to the related time block before this deadline, their reputation will be adversely affected due to the failure to create the new block. Figure 3.2.13 illustrates the case when a BSM data block is failed to created. As a result, there will exist a time block that does not have any BSM data block using its hash. The next BSM data block will be created based on a new time block in the time chain. The leader responsible for the failure to create the previous BSM data block can be identified by examining the time block and the preceding BSM data block.

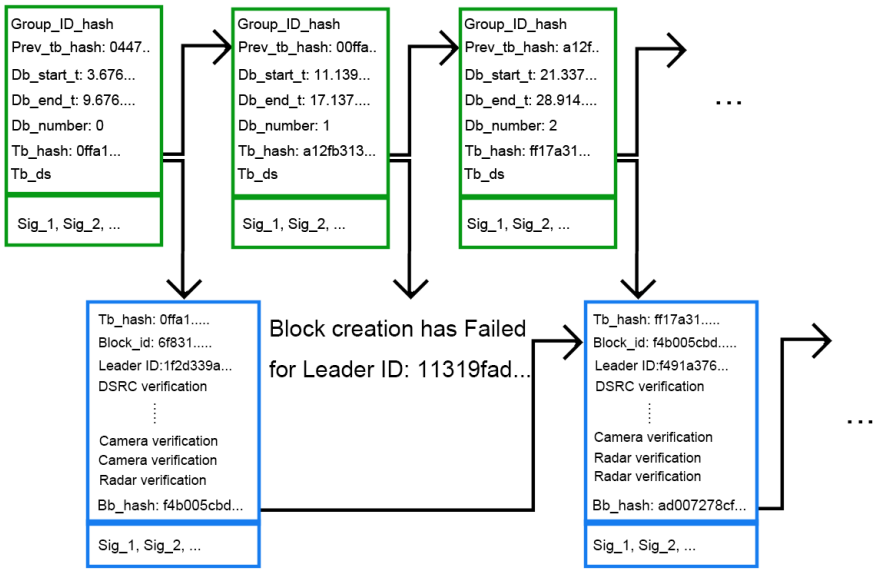

Fig. 3.2.13: Failure to Create a BSM Data Block: An Example.

After the BSM data block is created, the process returns to the first step in the

PoPr consensus: reputation assessment and consensus group member assignment. This cyclical action facilitates the creation of new BSM blocks for traffic data collection. Users can choose to rely solely on BSMs from contributors with high reputations, thereby ensuring the authenticity of the traffic data.

3.3 Threat model and Security Analysis

A threat model is crucial for assessing the security framework of a blockchain system. For the system we propose, it is hypothesized that adversaries have the capability to delay, internally discard, or introduce erroneous or fraudulent messages. Furthermore, they may alter messages or inundate the system with spurious or fraudulent communications. Additionally, it is postulated that these adversaries might collaborate with compromised CAVs, and they could generate an indefinite number of virtually fake nodes.

For the consensus group, it is assumed that the adversary can control at most $\lfloor \frac{(G-1)}{3} \rfloor$ members.

Presented here is an analysis of various potential attacks, conducted by the assumed adversaries, against our proposed system:

3.3.1 DDOS Attack

A DDOS attack can be launched using multiple sources to overwhelm the target nodes. In our system, members of the consensus group and the leader are particularly vulnerable to such attacks. To mitigate this threat, the blockchain system can be designed to incorporate additional group members when a DDoS attack is detected. If the leader becomes the target of a DDoS attack, they could opt to forfeit the opportunity to create the new BSM block, allowing the next leader in the group to assume responsibility for the block creation. Furthermore, the inherent design of the consensus group, which is based on reputation, naturally reduces the effectiveness of DDoS attacks. This is because the membership of the consensus group changes with the creation of each new BSM data block. As a result, adversaries would need to

constantly reconfigure their attacks, diminishing the overall effectiveness of the DDoS attack, especially when there are numerous or frequently changing targets.

3.3.2 Spoofing Attack

When a spoofing attack occurs, adversaries attempt to impersonate existing CAVs, disseminating fake BSMs or BSM verification messages. However, the digital signature scheme prevents this by ensuring that adversaries cannot forge a message without access to the private key. The computational difficulty of reverse-computing the private key from the public key and digital signature is prohibitive for classical computers. In the event that adversaries possess quantum computing capabilities, which could potentially enable the reverse computation of private keys, the implementation of a post-quantum digital signature scheme would effectively safeguard against this threat. Consequently, the spoofing attack can be mitigated.

3.3.3 Sybil Attack

Sybil attacks occur when an adversary attempts to create multiple fake CAVs using legitimate digital signature key pairs. This type of attack is particularly powerful against traditional PoR blockchains. If the reputation gain mechanism is not properly designed, the adversary could generate an infinite number of nodes, especially if these fake nodes are not effectively detected by the blockchain. Methods to mitigate this attack in PoR blockchains include combining PoR with PoW [26, 30], although this may sacrifice the system's real-time capabilities. Alternatively, a limited-scale approach can be used to maintain real-time performance while also limiting the damage caused by Sybil attacks [10]. In our proposed approach, nodes in the VANET are verified through verification messages, and each node must prove its "presence" to gain reputation. This significantly raises the cost for an adversary to launch a Sybil attack, as non-existent nodes cannot be verified by CAVs on the road. Additionally, faking camera or radar data to artificially increase reputation requires substantial computational resources, as producing highly authentic camera images is challeng-

ing. However, legitimate nodes in the system can easily increase the rigor of the authentication process if they detect that some camera images are fake.

Nevertheless, there are two scenarios in which a Sybil attack could be successfully launched against our system. These scenarios are discussed in the following sections.

3.3.4 Brute-force Attack

A brute-force attack targets nodes with simple IDs and weak passwords. It can also exploit systems that do not implement a secure password design. A simple example would be a system that uses only an 8-digit password or a public key without limiting password attempts. In such a case, an adversary could brute-force the password with 10^8 attempts. In our proposed system, the digital signature scheme incorporates a specific key pair generation algorithm to ensure that the key is difficult to brute-force. The key size is at least 512 bits, meaning an adversary would need to perform up to 2^{512} computations to find the private key, making a brute-force attack infeasible.

3.3.5 GPS Spoofing Attack

In our system, GPS spoofing can be mitigated using BSM verification messages. When adversary nodes attempt to use faulty GPS data in their BSMs, nearby honest CAVs can verify the authenticity of the GPS information. If the GPS data is found to be false, a BSM faulty message is generated in the BSM data block, which subsequently decreases the reputation of the adversary-controlled nodes.

3.3.6 Message Fabrication Attack

If an adversary attempts to launch a message fabrication attack, they must contend with the hash included in the message, as the hash generated by a hash function is unique for every message. If the attacker tampers with a message sent by other nodes, the hash will not match when the nodes verify it against the provided input in the message. The most effective method to fabricate a message containing a hash would be to find a collision in the hash present in the message. However, finding

such a collision would require at least 2^{128} attempts for a 256-bit hash, exploiting the birthday paradox [13], which makes a message fabrication attack infeasible.

3.3.7 Group Consensus Attack

This attack assumes that at least one node in our consensus group is controlled by an adversary. The adversary could vote in favor of a faulty BSM data block if it is generated by another consensus group member also under their control. However, given the assumption that only $\lfloor \frac{(G-1)}{3} \rfloor$ members can be controlled by the adversary, they will not have enough votes to successfully validate the block, as the number of votes required to pass consensus is greater. In practical settings, this attack can be mitigated by introducing additional group members if some members are behaving suspiciously.

3.3.8 "Go Bullying" Attack

This variant of the Sybil attack is specifically tailored to our proposed system. Figure 3.3.1 illustrates this attack. In general, the adversary needs to physically drive some CAVs on the road to block the other CAVs' access to the CAV marked in red. In this scenario, the CAV in red could either exist or not exist, as no other CAVs, except those controlled by the adversary, can verify its presence.

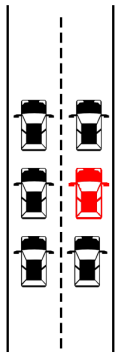

Fig. 3.3.1: Go Bullying Attack.

However, this attack is challenging to execute due to the complexity of real traffic

conditions. Additionally, RSUs can help mitigate this attack, as they can detect whether the red CAV exists when other CAVs pass by them.

3.3.9 Isolate Region Attack

This describes another variant of the Sybil attack tailored for our system, as illustrated in Figure 3.3.2. In this scenario, the gray and blue regions are isolated and not accessible to each other. Consequently, any BSM or BSM verification message within the blue or gray region is entirely controlled by the nodes inside that specific region. This isolation can lead to a fork in our blockchain because group members in either the gray or blue regions are unable to verify new BSM data blocks contains BSM verification messages from the other region. To initiate the attack, the adversary may create or declare a closed region that is inaccessible to other CAVs. Within this secluded area, the adversary can generate an indefinite number of real or virtual nodes. These nodes, by verifying each other, can artificially boost their reputation within the blockchain.

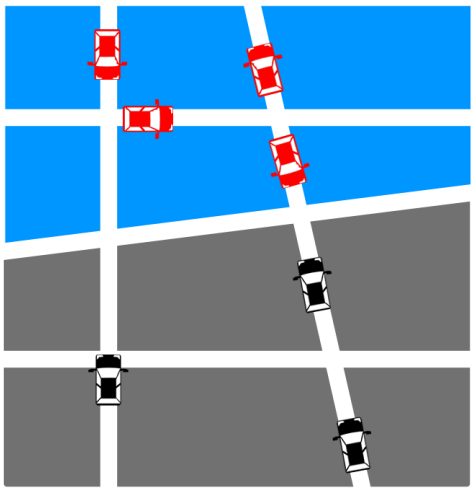

Fig. 3.3.2: Isolation Region Attack.

To mitigate this attack, we can employ a fork chain approach, which splits the blockchain into two: one for nodes within the isolated region and another for verify-

ing CAVs in the open region accessible to all nodes. In this scenario, honest nodes and users can choose to join the trustworthy blockchain, leaving the adversary to operate a separate blockchain devoid of honest participation. This isolation reduces the adversary's potential targets and diminishes the impact of the attack.

3.4 Summary

In this chapter, we introduced the PoPr blockchain system, starting with the elements that comprise the system in the context of VANET. We then detailed the architecture of the PoPr blockchain system, including the types of verification messages, the two blockchain types utilized in the system, and the PoPr consensus algorithm. Finally, we analyzed the security of the PoPr blockchain system by examining the effects of various attacks on the system.

The PoPr blockchain system is relatively robust against most attacks designed for VANET, making it a viable option for implementing a secure traffic data collection system. The performance of the system will be discussed in the next chapter.

CHAPTER 4

Simulation Results

To test the PoPr blockchain system, we need a VANET along with CAVs and RSUs equipped with multiple sensors. Conducting a real-world experiment is impractical, as it would require extensive resources, including controlled streets and the necessary CAVs and RSUs. Thus, we have decided to build a simulator to run the PoPr blockchain system. However, given the scale of the VANET and blockchain system, implementing all aspects of PoPr in a simulator is also very impractical. As a result, we plan to focus on simulating key components of the PoPr blockchain system, specifically prioritizing the system's feasibility during the implementation of the simulator. However, due to the simulator's design, we cannot directly test performance metrics such as throughput or latency. Therefore, we will use the results obtained from the simulator, along with assumptions about several related parameters, to estimate the system's performance.

4.1 Setup

4.1.1 CARLA Simulator

To simulate real-world sensor data, the CARLA simulator is a practical choice for emulating CAVs and RSUs within the PoPr blockchain system. The CARLA simulator is an open-source platform designed for research in autonomous driving. It creates a dynamic environment and offers an easy-to-use interface for an agent to interact with this environment. It is structured as a server-client system, with the server managing the simulation and rendering the scene. The client API, implemented in Python, fa-

cilitates communication between the autonomous agent and the server using sockets [31]. Figure 4.1.1 displays a scene captured while the CARLA simulator is running, showing CAVs randomly driving on the street.

Fig. 4.1.1: CARLA Simulator Depicting CAVs in Operation.

However, given that CARLA is built on Unreal Engine 4—a graphics engine that demands significant computational resources—it requires optimization to run effectively. Due to time constraints in implementing the simulator and conducting the experiments, we have optimized CARLA to some extent. This allows it to operate in an acceptable state for generating the necessary sensor data.

4.1.2 Docker

Although the network simulator NS-3 offers superior capabilities for simulating network interfaces, it does not natively support Python, the programming language we use for implementing most functions of the PoPr blockchain. Given the time constraints, it is not feasible for us to develop the PoPr simulator based on NS-3's interface. Alternatively, we have chosen to use Docker to manage the network interfaces for nodes in the VANET. Docker is an open platform designed for developing, shipping, and running applications. It allows us to package and run an application within a loosely isolated environment known as a container. This isolation and security enable us to run multiple containers simultaneously on a single host [32]. Docker provides a subnet interface, enabling containers to communicate with each other using

subnet IPs. This capability allows us to simulate the transport layer of DSRC and C-V2X effectively. Specifically, it facilitates the use of TCP and UDP protocols for communication between containers, using customized message protocols. Therefore, we can design various types of messages for use in the PoPr blockchain system and facilitate their exchange between Docker containers.

4.1.3 Simulator Architecture

The architecture of the simulator is illustrated in Figure 4.1.2. The simulator is divided into two components due to the limitations of the CARLA simulator and time constraints. The CARLA simulator emulates CAVs and RSUs. After a specified time interval, it pauses the simulation and sends sensor results via UDP messages. These sensor results are transmitted through the Docker interface, which allows Docker to redirect the messages to each respective container. Once the container that simulates CAVs and RSUs receives the sensor results, it will encapsulate these results into BSMs and BSM verification messages. These messages are then sent via TCP and UDP to the appropriate receivers to simulate the PoPr consensus process. After a designated interval, the CARLA simulator resumes operation to gather sensor data for the next cycle. Meanwhile, the Docker containers that have completed processing the PoPr consensus will await the arrival of new sensor data delivered by CARLA.

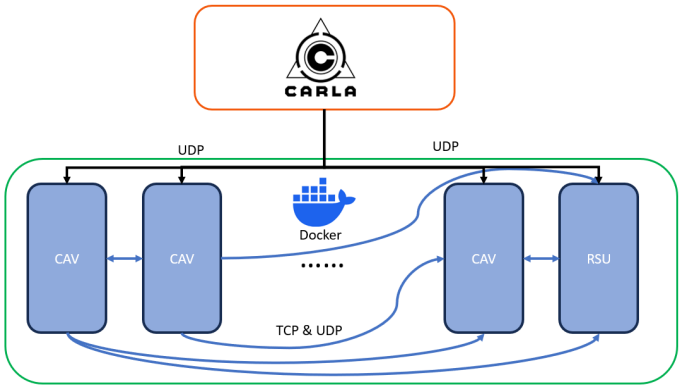

Fig. 4.1.2: Simulator Architecture.

This simulator design allows us to obtain comprehensive results for the blockchain and the reputation of nodes. However, it reduces the feasibility of testing the real-

time aspects of the blockchain system. Nonetheless, this design significantly lowers the hardware overhead required to run the simulation, which is crucial for its successful execution. Additionally, given that communication between CARLA and Docker is one-sided, this approach provides a high tolerance for the synchronicity of the two components.

4.1.4 Simulation Setting

The simulation parameters for the CARLA, Docker, and PoPr blockchain systems are provided in the following tables:

Parameters	**Values**
CARLA Simulator Version	0.9.15
Simulation Time	25 seconds
Processing Time	4 seconds
Docker Container Running Time	15 seconds
CAV Type	Nissan Micra
CAV Amount	31 units
RSU Amount	3 units
Camera Sensor Amount	2 units, front and rear
Camera Resolution	800×600
Radar Sensor Amount	2 units, front and rear
Radar Range	30 meters
Radar Angle	90 degrees, front and rear
Radar Scanning Rate	4000 points/s
GNSS	Yes

Table 4.1.1: CARLA Simulation Pararmeters.

Table 4.1.1 displays the parameters used in the CARLA simulator for the simula-

tion. The *Simulation time* refers to the duration CARLA takes to run the simulation on a frame-by-frame basis. Due to the heavy computational load of the simulation, a 25-second run in real time equates to only 4 seconds of simulated time within the simulator. The *Processing Time* refers to the duration required for CARLA to process sensor data and prepare it for transmission to Docker. The *Docker Container Running Time* is the interval during which CARLA waits for the nodes to execute after sending the sensor data to the Docker containers.

Parameters	**Values**
Container Amount	34 units
Network Type	Bridge
Subnet IPs	192.168.5.0/24
Volume Used	Yes

Table 4.1.2: Docker Simulation Pararmeters.

Table 4.1.2 displays the parameters used in Docker. The network type 'Bridge' enables the containers to communicate within the subnet. The shared volume allows containers to save and access the consensus results.

Parameters	**Values**
VANET Protocols	TCP, UDP
BSM Interval	4 seconds
DSRC Verification Weights	0.5
Camera Verification Weights	1.0
Radar Verification Weights	2.0
BSM Verify Weights: DSRC	0.6
BSM Verify Weights: Camera	1.2
BSM Verify Weights: Radar	2.4
Consensus Group Member size	8
max_t	8
Decay Coefficient	0.15
Sigmoid Coefficient	[35, 2]
Degree of Tolerance: Angle	2
Degree of Tolerance: Depth	2
$base\_t\_vote$	4
$max\_t\_vote\_mod$	3
Max BSM Block Creation Time	8

Table 4.1.3: PoPr Blockchain Simulation Pararmeters.

Table 4.1.2 displays the parameters used in the simulation of the PoPr blockchain system. Note that the weight assignments for BSM fault messages and faulty leaders are not provided, as these features were not implemented due to time constraints. Additionally, the machine learning model for processing camera images is not included due to time constraints in this thesis. Instead, the camera detection results are directly obtained from the CARLA simulator. Moreover, Dilithium2 is used as the digital signature scheme in the PoPr blockchain simulation.

4.1.5 Evaluation Metrics

To evaluate the performance of a blockchain system, throughput and latency are two crucial measurements. Throughput refers to the number of transactions a blockchain can process within a given period, while latency is the time it takes for a transaction to be confirmed and added to the blockchain. For the PoPr blockchain system, throughput is measured by the number of BSM verification messages confirmed in a block within a certain period, and latency refers to the time it takes for a BSM verification message to be confirmed. Given that the entries in the BSM data blocks consist of verification messages, the unit of measurement for throughput in the PoPr blockchain system is defined as Messages Per Second (MPS). The equation used to calculate throughput is presented below:

$$MPS = \frac{Number\ of\ Verification\ Message\ in\ the\ Block}{Block\ Creation\ Time}$$

Conversely, latency in the system is measured in seconds. Latency is defined as the time elapsed from when a verification message is submitted to the consensus group members until the message is confirmed in a new BSM data block. The equation used to calculate latency is presented below:

$$Latency = Message\ Confirmed\ Time - Message\ Submission\ Time$$

Other meaningful performance indicators include the average change in reputation related to the size of the BSM data blocks. This metric reflects the impact of data block size on the reputation of nodes. Additionally, the block size of the BSM data blocks can illustrate the transmission overhead associated with the blockchain in a VANET. The reputation impact on target nodes during a "Go bullying" attack is also a meaningful measurement. This metric is particularly significant, as most other attacks do not directly affect the reputation of nodes in the PoPr blockchain system.

4.1.6 Simulation Setup and System Information

All simulator codes are implemented and executed in the following environment:

- OS: Windows 11, Windows Subsystem for Linux 2
- Processor: Intel i9-14900k 3.20 GHz
- Memory: 128 GB
- GPU: Nvidia RTX 3080ti

The experiments were conducted using the following programming languages and libraries:

- Programming Language: Python 3.10.14
- Library: Carla 0.9.15, Cv2, Json, Hashlib, Merkletools, Numpy, Oqs, Select, Socket, Threading

4.2 Performance Evaluation

We conducted the PoPr blockchain simulation under specific simulation settings. The longest blockchain generated during the simulation consisted of 21 blocks. For the purposes of performance evaluation, we focused on analyzing the first 10 blocks as our simulation dataset. Additionally, we compared the throughput and latency of the PoPr approach, as well as resilience in extreme attack scenarios, with those of PoW and PoE outlined in [18]. For PoW, we assume the block generation rate in PoW systems is 10 minutes per block. This assumption is based on the behavior of Bitcoin, which employs PoW and adjusts the timing of new block generation to approximately every 10 minutes, based upon the difficulty of the *target* hash [33]. The block generation time in PoW systems cannot be significantly shortened without consequences, as doing so would increase the likelihood of blockchain forks. This results in more orphan blocks and reduces the overall stability of the network. For

PoE, we will compare our results with the 1-of-6 voting policy in PoE, which exhibits the best latency and throughput within their system.

The throughput for each of the first 10 blocks, which involve 34 nodes, is detailed in Figure 4.2.1. This figure compares the throughput of our approach with those of the PoW and PoE methods.

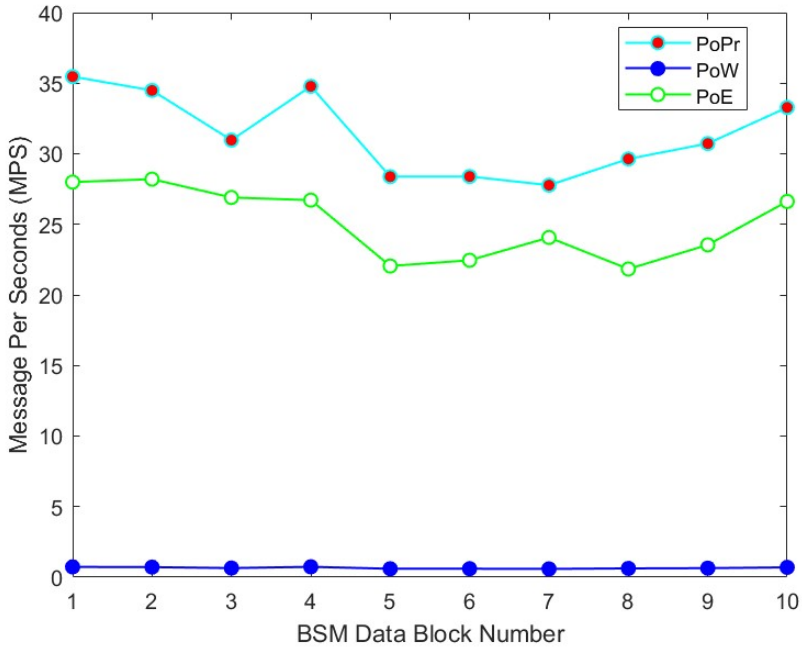

Fig. 4.2.1: Comparison of Throughput: PoPr, PoE, and PoW Approaches

Throughput is measured by the block creation time for each block. Specifically, the time interval is calculated from the end of the previous block's creation to the commencement of the current block's creation. The average time required to create a BSM data block is approximately 12.57 seconds, assuming each block is instantly created after reaching the designated end time, Db_end_t. According to Figure 4.2.1, the PoPr approach has the highest MPS rate observed among the first 10 blocks is 35.47 MPS, while the lowest is 27.77 MPS. Given that only 34 nodes are present in the blockchain network, this throughput level is deemed adequate to efficiently handle all verification messages from the nodes. Additionally, both our method and PoE significantly outperform the PoW approach, which has a maximum MPS rate of only 0.735 MPS and a minimum of 0.588 MPS. This lower performance in PoW is

due to its requirement of approximately 10 minutes to initiate consensus, rendering it unsuitable for real-time tasks compared to the more responsive PoPr and PoE approaches.

The latency of the first 10 blocks, involving 34 nodes, across three different approaches is presented in Figure 4.2.2. Due to the fact that PoW requires at least 598 seconds to incorporate a new message into a block, while PoPr and PoE require less than 6 seconds, Figure 4.2.2 primarily illustrates the significant latency disparity between PoW and the faster PoPr and PoE approaches. The average latency for PoW is 599.32 seconds, rendering it unsuitable for real-time tasks such as traffic data collection.

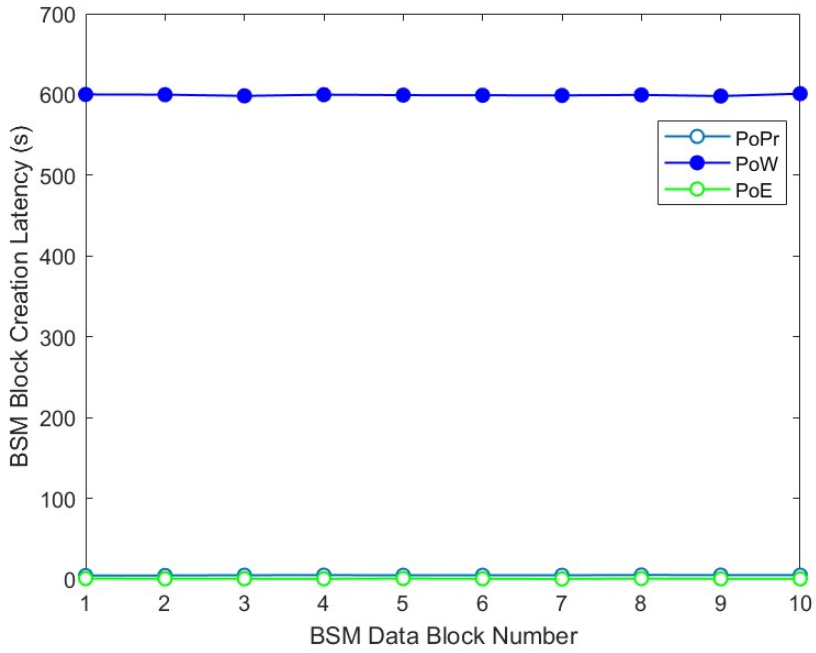

Fig. 4.2.2: Comparison of Latency: PoPr, PoE, and PoW Approaches.

Figure 4.2.3 demonstrates the latency of the PoPr and PoE approaches. The average latency for PoPr across the first 10 blocks is 4.78 seconds. The minimum latency observed within these blocks is 4.43 seconds, noted in the first BSM data block, while the maximum latency recorded is 5.06 seconds. Conversely, the average latency for PoE is 0.454 seconds, with a minimum latency of 0.1978 seconds and a maximum of 0.7396 seconds. This efficiency is attributed to the simplicity of the PoE consensus

algorithm, where the leader, either a CAV or RSU, can be instantly determined using an election algorithm based on the bully algorithm [18]. However, this algorithm is vulnerable to inconsistency and delay attacks, as an adversary could manipulate the system by faking delay times when sending a leader request. Consequently, our PoPr approach remains a secure option, as the maximum latency is not excessively prolonged and is sufficiently short for a traffic data collection system. This ensures the system's real-time capabilities are maintained.

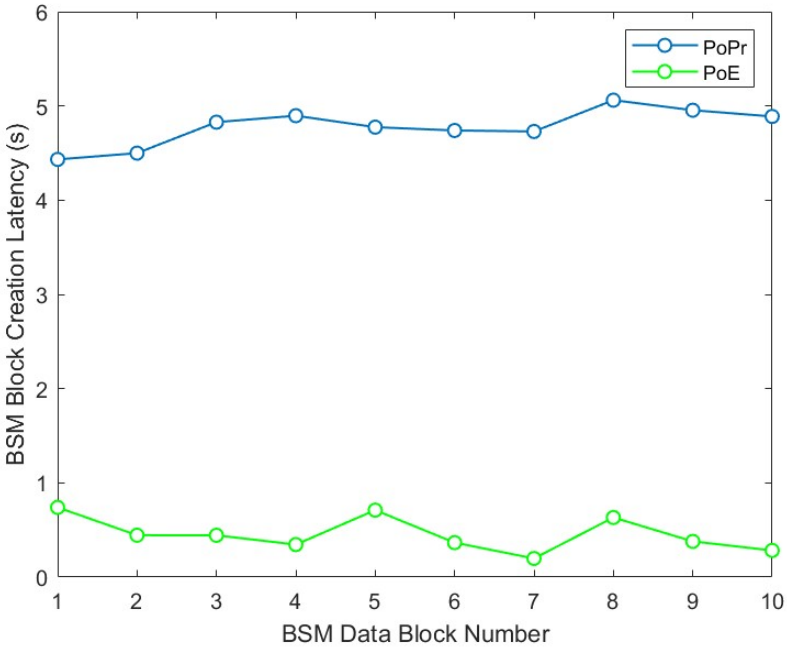

Fig. 4.2.3: Latency Comparison: PoPr vs. PoE Approaches.

In Figure 4.2.4, the distribution of three types of verification messages is presented. Unexpectedly, DSRC verification messages significantly outnumber those from camera and radar systems. The proportion of the three types of verification messages is approximately 1:0.1:0.05. This distribution aligns with the range of different sensors available to BSMs.

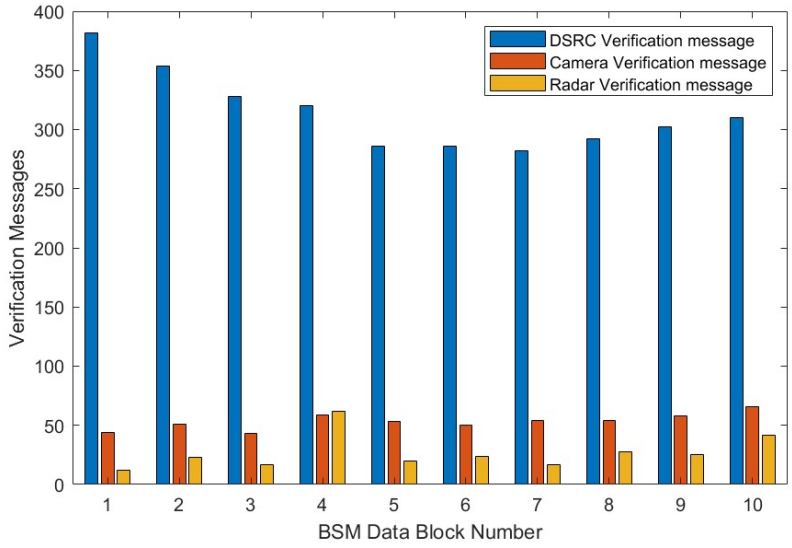

Fig. 4.2.4: Distribution of Three Types of Verification in the PoPr Blockchain.

Figure 4.2.5 illustrates the block size of each BSM data block when using Dilithium and ECDSA digital signature schemes. The average block size in the PoPr Blockchain with Dilithium is considerably large, at 4149 KB. This size is primarily due to the digital signatures generated by Dilithium2, which are approximately 2.4 KB each. To address this overhead, one option is to switch to a digital signature scheme with smaller signatures, such as ECDSA. The average size of a BSM data block with ECDSA as digital signature is 491 KB, which is about 9 times smaller than when using Dilithium. Consequently, it is crucial to identify scenarios where specific security needs dictate the choice between ECDSA and Dilithium, as each offers distinct advantages. Alternatively, without compromising the security of digital signatures, adopting a 5G NR Uu interface in C-V2X could enhance data transmission rates to better accommodate larger data blocks. The advanced capabilities of 5G could offer CAVs improved data transmission rates, facilitating the efficient delivery of messages and data blocks.

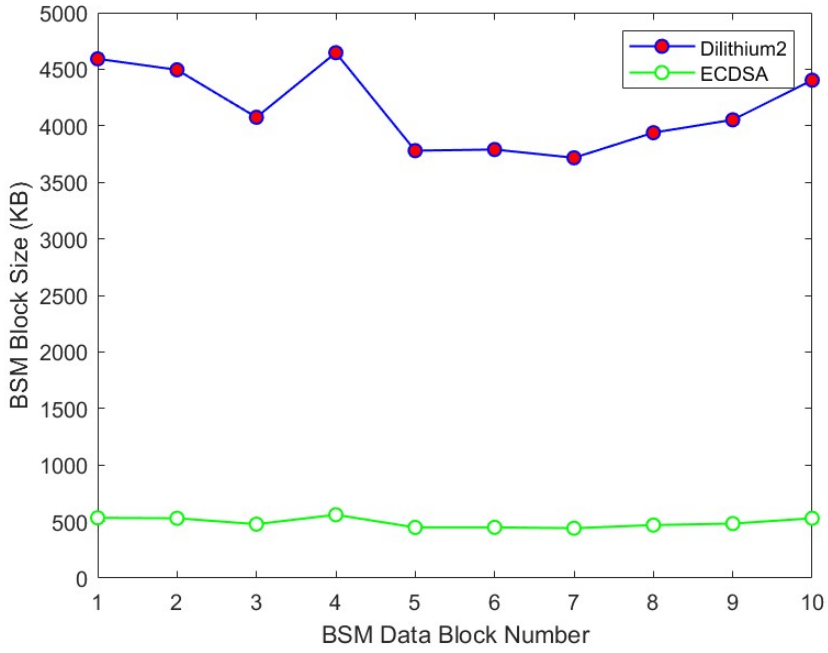

Fig. 4.2.5: BSM Data Block Size in the PoPr Blockchain.

The relationship between BSM data block size and the average reputation of the first 10 blocks, involving 34 nodes, is illustrated in Figure 4.2.6. The highest average reputation, observed in block 4, is 0.20, while the lowest average reputation, observed in block 9, is 0.06. Comparing the average reputation of block 1 with block 4, we found that the size of the verification messages is approximately the same. However, the reputation associated with block 4 is significantly higher than that of block 1. This difference can be attributed to the composition of the verification messages within the blocks. Block 4 contains a higher proportion of verification messages from camera and radar systems, which contribute to a higher reputation, whereas block 1 predominantly includes DSRC verification messages.

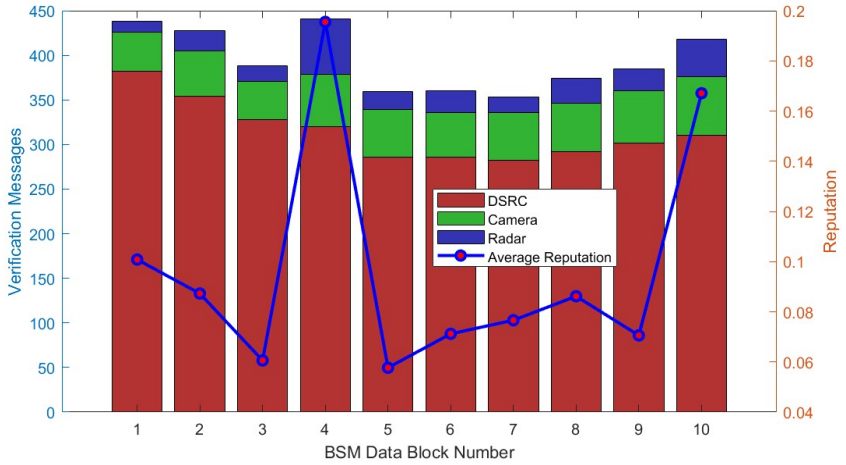

Fig. 4.2.6: BSM Data Block Size versus Average Reputation.

In Figure 4.2.7, the relationship between the number of adversary-controlled nodes and the attack success rate without lurking strategy in the blockchain system is illustrated. This represents an extreme case of attack, with the adversary directly controlling nodes within the blockchain. For our system, the maximum number of nodes that an adversary can control is capped at 34. Unfortunately, due to time constraints, we are unable to include a comparison with the PoE approach in this analysis. Instead, we present the comparison of attack scenarios for PoW and PoPr. The attack success rate for a PoW is defined as the probability that adversary-controlled nodes can extend the blockchain by successfully gaining control over the majority of the network's nodes. For PoPr, it is defined as the probability of the adversary controlling at least $\lceil \frac{2 \cdot (G-1)}{3} \rceil$ group members of the consensus group.

From the figure, it is evident that the attack success rate exceeds 50% once the adversary controls 18 nodes in a PoW blockchain. This finding aligns with the security threshold for a 51% attack as recognized in Bitcoin's design [12]. Conversely, in the PoPr approach, the attack success rate remains at 0% until the adversary controls more than 30 nodes. This resilience is due to our design ensuring that nodes exhibiting dishonest behavior cannot enter the consensus group. Consequently, an adversary would need to control almost all nodes in the blockchain to control the new block generation.

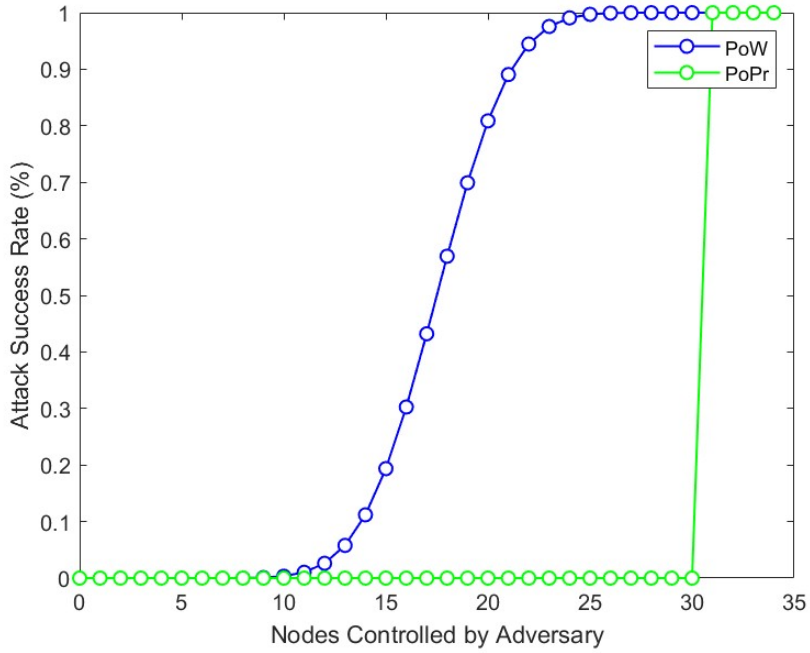

Fig. 4.2.7: Adversary Control of Nodes vs. Successful Attack Rate: No Lurking Phase.

Alternatively, Figure 4.2.8 depicts a scenario where the adversary adopts a lurking strategy, initially behaving in a manner consistent with PoPr protocols, and subsequently initiating the production of faulty messages. In this scenario, every node controlled by the adversary has the potential to become part of the consensus group, thereby increasing the likelihood of a successful attack. Our PoPr approach exhibits a slightly higher vulnerability compared to PoW, with the attack success rate exceeding 50% once the adversary controls 16 nodes. However, it outperforms PoW when the adversary controls more than 20 nodes. To mitigate this increased risk of attack in PoPr, expanding the size of the consensus group could be an effective strategy.

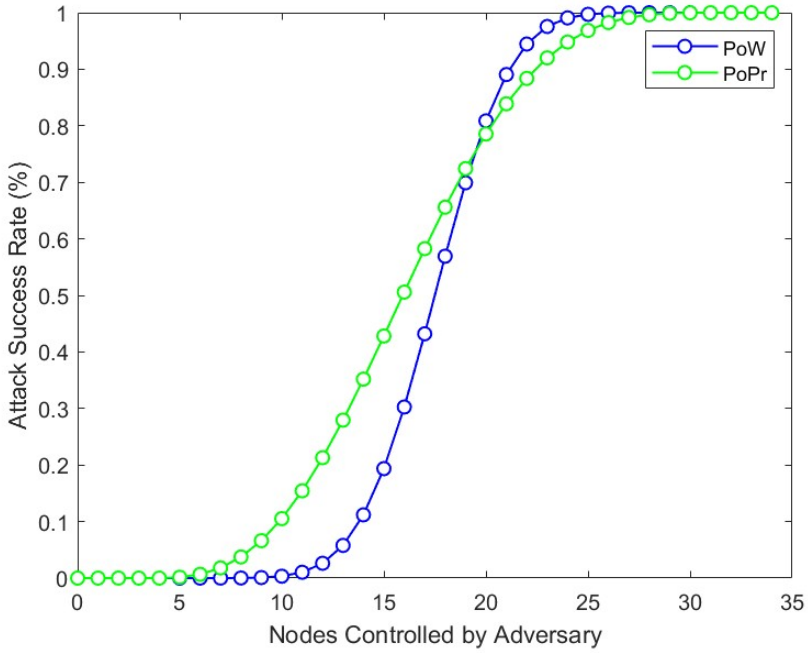

Fig. 4.2.8: Adversary Control of Nodes vs. Successful Attack Rate: Lurking Phase Applied.

Finally, the estimation of the "Go Bullying" attack's impact on reputation in the PoPr system is presented in Figure 4.2.9. The target CAVs are those that adversaries aim to exploit by faking verification messages to falsely confirm the "presence" of the target CAVs. As observed in the figure, the reputation of these targeted CAVs increases linearly. This growth is predicated on the assumption that adversary-controlled CAVs dispatch all three types of verification messages for target CAVs. Additionally, it is assumed that the adversary would need at least six CAVs to completely surround a target CAV. Although the reputation of the target CAVs appears to grow rapidly as more are introduced into the network, the cost of the attack also increases significantly. For example, to falsely claim the "presence" of 16 target CAVs on the road, the adversary would need to deploy at least 96 CAVs working collaboratively to create the necessary surrounding spaces. This approach is neither efficient nor feasible as a method of attack. Moreover, as more honest nodes continue to join the PoPr network, the effectiveness of the attack diminishes.

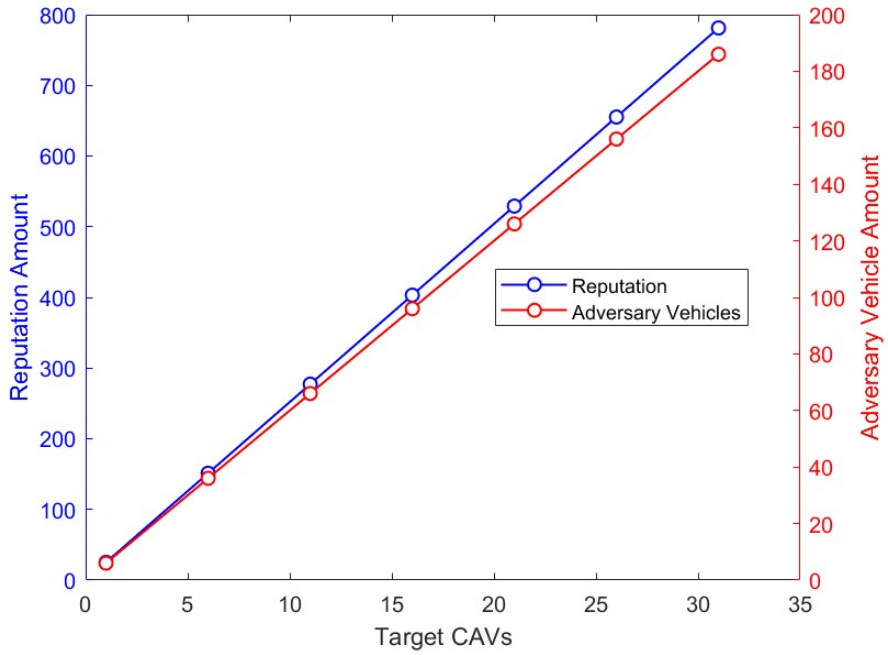

Fig. 4.2.9: The Impact of the "Go Bullying" Attack on Reputation.

4.3 Summary

In this chapter, we introduced the setup and structure of the PoPr blockchain simulator, along with the simulation settings used. We illustrate the simulation results through various figures, which include analyses of throughput, latency, and the effects of three types of verification messages. We also compare the performance of PoPr approach with PoW and PoE in terms of throughput, latency, and resilience under extreme attack scenarios where the adversary can directly control nodes.

From the simulation results, we observed that the throughput and latency metrics of the PoPr block demonstrate relatively well performance compare with PoW and PoE, effectively maintaining the real-time operation of the system. However, the use of Dilithium significantly increases the size of the BSM data blocks, introducing considerable overhead and necessitating a trade-off between security and performance. Furthermore, we explored the relationship between reputation and both the size and types of verification messages contained within the BSM data blocks. We also found

that the PoPr approach is resilient against adversaries controlling a considerable number of nodes without employing a lurking strategy. However, its security is relatively compromised when adversaries control nodes and apply a lurking strategy.

CHAPTER 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we present a blockchain-based VANET system designed for traffic data collection. We have developed a new consensus algorithm, PoPr, aimed at reducing the overhead associated with PoW. Through our security analysis, we determined that PoPr is capable of defending against both VANET-specific and general blockchain system attacks. Furthermore, we identify two attacks that could specifically target our PoPr blockchain system: the "Go Bullying" Attack and the Isolate Region Attack. We also propose feasible solutions to mitigate these attacks.

For the simulation to test the performance of the PoPr Blockchain system, we employed a separate simulator approach. This approach involved using the CARLA simulator, which provides sensor data, and Docker, which offers the network capabilities required to emulate the VANET. We divided the simulator into two components, comromising some aspects of real-time performance to optimize hardware efficiency and ensure the simulator could run effectively.

From the simulation results, we observed that the throughput and latency metrics of the PoPr block demonstrate solid performance compared to PoW and PoE, effectively maintaining the system's real-time operation. However, the size of the BSM data blocks introduces considerable overhead, requiring a trade-off between security and performance. Moreover, the PoPr approach shows relative vulnerability when facing a strong adversary capable of controlling nodes and executing a lurking attack. Additionally, the relationship between reputation and BSM data block characteristics

was explored.

5.2 Future Work

Due to time and hardware constraints, we were unable to implement and explore several aspects of the PoPr Blockchain in this thesis. These include the design of BSM faulty messages and the development of an object detection algorithm for camera verification messages. Future work could focus on addressing these aspects and testing the system with a simulator that emphasizes real-time capabilities. Additionally, the verification methods used to prove the "presence" of CAVs could be expanded to cover other dimensions, thereby making the verification process more varied, efficient, and reliable.

REFERENCES

[1] G. Leduc *et al.*, "Road traffic data: Collection methods and applications," *Working Papers on Energy, Transport and Climate Change*, vol. 1, no. 55, pp. 1–55, 2008.

[2] J. B. Kenney, "Dedicated short-range communications (dsrc) standards in the united states," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[3] M. S. Anwer and C. Guy, "A survey of vanet technologies," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 5, no. 9, pp. 661–671, 2014.

[4] G. Twardokus and H. Rahbari, "Evaluating v2v security on an sdr testbed," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2021, pp. 1–3.

[5] M. H. C. Garcia, A. Molina-Galan, M. Boban, J. Gozalvez, B. Coll-Perales, T. Şahin, and A. Kousaridas, "A tutorial on 5g nr v2x communications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1972–2026, 2021.

[6] H. Hasrouny, A. E. Samhat, C. Bassil, and A. Laouiti, "Vanet security challenges and solutions: A survey," *Vehicular Communications*, vol. 7, pp. 7–20, 2017.

[7] M. Azees, P. Vijayakumar, and L. Jegatha Deborah, "Comprehensive survey on security services in vehicular ad-hoc networks," *IET Intelligent Transport Systems*, vol. 10, no. 6, pp. 379–388, 2016.

[8] M. Saad, M. K. Khan, and M. B. Ahmad, "Blockchain-enabled vehicular ad hoc

networks: A systematic literature review," *Sustainability*, vol. 14, no. 7, p. 3919, 2022.

[9] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," *arXiv preprint quant-ph/0301141*, 2003.

[10] 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*, vol. 6, no. 2, pp. 177–186, 2020.

[11] R. Zhang, R. Xue, and L. Liu, "Security and privacy on blockchain," *ACM Computing Surveys (CSUR)*, vol. 52, no. 3, pp. 1–34, 2019.

[12] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[13] J. Liu, "Digital signature and hash algorithms used in bitcoin and ethereum," in *Third International Conference on Machine Learning and Computer Application (ICMLCA 2022)*, vol. 12636. SPIE, 2023, pp. 1302–1321.

[14] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "Crystals-dilithium: A lattice-based digital signature scheme," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 238–268, 2018.

[15] J. Shen, D. Liu, X. Chen, J. Li, N. Kumar, and P. Vijayakumar, "Secure real-time traffic data aggregation with batch verification for vehicular cloud in vanets," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 807–817, 2019.

[16] Y. Dieudonné, B. Ducourthial, and S. M. Senouci, "Col: A data collection protocol for vanet," in *2012 IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 711–716.

[17] Y.-T. Yang, L.-D. Chou, C.-W. Tseng, F.-H. Tseng, and C.-C. Liu, "Blockchain-based traffic event validation and trust verification for vanets," *IEEE Access*, vol. 7, pp. 30 868–30 877, 2019.

[18] H. Guo, W. Li, M. Nejad, and C.-C. Shen, "Proof-of-event recording system for autonomous vehicles: A blockchain-based solution," *IEEE Access*, vol. 8, pp. 182 776–182 786, 2020.

[19] C.-H. Lo, W.-C. Peng, C.-W. Chen, T.-Y. Lin, and C.-S. Lin, "Carweb: A traffic data collection platform," in *The Ninth International Conference on Mobile Data Management (mdm 2008)*. IEEE, 2008, pp. 221–222.

[20] N. Caceres, L. M. Romero, F. G. Benitez, and J. M. del Castillo, "Traffic flow estimation models using cellular phone data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 3, pp. 1430–1441, 2012.

[21] K. A. Yadav and P. Vijayakumar, "Vanet and its security aspects: a review," *Indian Journal of Science and Technology*, vol. 9, no. 18, pp. 104–118, 2016.

[22] J. Ren and D. Xia, "Introduction to 5g c-v2x," in *Autonomous driving algorithms and Its IC Design*. Springer, 2023, pp. 283–294.

[23] V. Mannoni, V. Berg, S. Sesia, and E. Perraud, "A comparison of the v2x communication systems: Its-g5 and c-v2x," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE, 2019, pp. 1–5.

[24] E.-h. Diallo, O. Dib, and K. Al Agha, "A scalable blockchain-based scheme for traffic-related data sharing in vanets," *Blockchain: Research and Applications*, vol. 3, no. 3, p. 100087, 2022.

[25] H. Chai, S. Leng, K. Zhang, and S. Mao, "Proof-of-reputation based-consortium blockchain for trust resource sharing in internet of vehicles," *IEEE Access*, vol. 7, pp. 175 744–175 757, 2019.

[26] Y. Wang, Z. Su, K. Zhang, and A. Benslimane, "Challenges and solutions in autonomous driving: A blockchain approach," *IEEE Network*, vol. 34, no. 4, pp. 218–226, 2020.

[27] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities," *IEEE access*, vol. 7, pp. 85 727–85 745, 2019.

[28] F. G. Mármol and G. M. Pérez, "Trip, a trust and reputation infrastructure-based proposal for vehicular ad hoc networks," *Journal of network and computer applications*, vol. 35, no. 3, pp. 934–941, 2012.

[29] O. Aluko and A. Kolonin, "Proof-of-reputation: an alternative consensus mechanism for blockchain systems," *arXiv preprint arXiv:2108.03542*, 2021.

[30] 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.

[31] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[32] Docker, "Docker overview," 2024, accessed: 2024-08-13. [Online]. Available: https://docs.docker.com/guides/docker-overview/

[33] J. A. Dev, "Bitcoin mining acceleration and performance quantification," in *2014 IEEE 27th Canadian conference on electrical and computer engineering (CCECE)*. IEEE, 2014, pp. 1–6.

VITA AUCTORIS

NAME:	Jiasong Liu
PLACE OF BIRTH:	Shijiazhuang, Hebei, China
YEAR OF BIRTH:	1995
EDUCATION:	University of Manitoba
	Winnipeg, Manitoba, 2022