electronics MDPI

Article

BeHarmony: Blockchain-Enabled Trustworthy Communication and Legitimate Decision Making in Multi-Party Internet of Vehicles Systems

Guodong Jin [1], Linyi Xu [2], Zihan Zhou [3], Qi Shi [1], Zihao Li [2], Hao Xu [1] and Yinuo Liu [4,*]

[1] College of Electronics and Information Engineering, Tongji University, Shanghai 201804, China; 2252540@tongji.edu.cn (G.J.); qishi@tongji.edu.cn (Q.S.); hao.xu@ieee.org (H.X.)
[2] CREATe Centre, School of Law, University of Glasgow, Glasgow G12 8QQ, UK; linyixu24@gmail.com (L.X.); zihao.li@glasgow.ac.uk (Z.L.)
[3] Information Hub of HKUST(GZ), The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China; zzh45472@gmail.com
[4] Department of Economic Statistics, Shanghai Customs College, Shanghai 201204, China
* Correspondence: liuyinuo@shcc.edu.cn

Abstract: The rapid development of the Internet of Vehicles using centralized systems faces significant challenges, including reliability and security vulnerabilities and high latency. This paper introduces a blockchain-enabled authentication and communication network for scalable IoV to enhance security, reduce latency, and relieve the dependency on centralized infrastructures. The network applies blockchain-enabled domain name services and mutual authentication for fault tolerance consensus, such as PBFT and RAFT, featuring a primary layer of road side units and edge servers for inter-vehicle communication and a sub-layer within each vehicle for intra-vehicle communication. The study evaluates various scenarios and assesses roadside unit availability based on random distribution along vehicle routes. This paper also discusses the legal issues involved in the proposed model, highlighting that the IoV system should be governed by a contract-based decentralized IoV system comprising both smart contracts and traditional contracts. This model offers a novel approach to developing a decentralized, secure, efficient, and ethical IoV ecosystem, advancing autonomous and reliable vehicular networks.

Keywords: Internet of Vehicles; ATS; smart contracts; legitimate decision-making; GDPR; regulation

check for updates

Citation: Jin, G.; Xu, L.; Zhou, Z.; Shi, Q.; Li, Z.; Xu, H.; Liu, Y. BeHarmony: Blockchain-Enabled Trustworthy Communication and Legitimate Decision Making in Multi-Party Internet of Vehicles Systems. *Electronics* **2024**, *13*, 3219. https://doi.org/10.3390/electronics13163219

Academic Editor: Fabio Grandi

Received: 28 June 2024
Revised: 7 August 2024
Accepted: 9 August 2024
Published: 14 August 2024

1. Introduction

In recent years, the emergence of the Internet of Vehicles (IoV) has garnered significant attention, particularly within the realm of smart transportation, in enhancing security, reducing latency, and minimizing reliance on centralized infrastructures [1]. IoV communication systems are typically classified into two primary categories: intra-vehicle and inter-vehicle communications, collectively known as Vehicle to Everything (V2X) communications [2]. Intra-vehicle communication encompasses interactions among sensors, on-board units (OBUs), and electronic control units (ECUs) within a vehicle. On the other hand, inter-vehicle communication involves transmissions between vehicles, facilitated by wireless communication modules like OBUs, and other entities such as road side units (RSUs). The scope of inter-vehicle communication includes vehicle to vehicle (V2V), vehicle to infrastructure (V2I), and vehicle to pedestrian (V2P) interactions.

Despite its promising potential, IoV is accompanied by several security challenges and vulnerabilities. Security threats in IoV span a range of issues, including denial-of-service (DoS)/distributed denial-of-service (DDoS) attacks, eavesdropping, impersonation, man-in-the-middle (MITM) attacks, spoofing, Sybil attacks for inter-vehicle communications, and eavesdropping, masquerading, injection, DoS, and message spoofing attacks for intra-vehicle communications [3,4]. These threats jeopardize the confidentiality, integrity, privacy, authentication, and availability of IoV systems.

In response to these challenges, decentralized technologies, exemplified by blockchain, are increasingly recognized for their potential to bolster security across various sectors [5,6]. As a fundamental shift away from centralized systems, blockchain technology enhances security by distributing data across a network, thereby reducing the risks of single points of failure and increasing resistance to tampering and eavesdropping attacks [7]. This approach has seen applications not only in communication networks, where it secures radio access networks [8], but also in creating more robust peer-to-peer ecosystems [9]. In addition, blockchain technology supports diverse applications across data sharing, data transactions, and copyright protection, owing to its inherent features of decentralization, immutability, and transparency [10–12]. Moreover, blockchain technology is leveraged in broader domains, such as managing complex systems and transactional frameworks, where the secure, transparent handling of data is paramount [13–17].

However, the potential of these decentralized solutions is often hindered by the absence of robust communication identity authentication mechanisms and secure, confidential communication protocols within IoV. The dynamic connectivity nature of IoV and its sensitivity to latency require that security measures not only protect data but also accommodate the system's operational demands without introducing undue complexity. Consequently, to fully leverage the benefits of blockchain and other decentralized technologies, IoV connections must be designed to adhere to specific requirements that ensure both security and efficiency.

To address these requirements effectively, a comprehensive communication identity management system is essential for facilitating peer discovery and the secure peer-to-peer routing of P2P connections in IoV. Additionally, the servers and services supporting such a system must align with specific criteria to ensure optimal performance [18]. IoV comprises both inter-vehicle and intra-vehicle networks, each with distinct communication levels. While non-sensitive communications should seamlessly traverse between these levels upon user request, the transfer of sensitive information from the intra-vehicle network to the inter-vehicle network raises privacy and security concerns. Therefore, it is imperative to implement separate identity management systems for these networks, ensuring a controlled and conditional exchange of information.

The advancement of IoV technology is not merely a technological issue but also a complex societal concern. Its technological improvements must comply with legal and ethical standards. Previous legal issues arising from autonomous driving technology have already drawn scholars' attention, such as privacy issues [19–21], security issues [22,23], conflicts of interest when right-of-way is contested [24], and legal liabilities in traffic accidents [25–27]. While we should actively embrace technological innovation, we must also recognize the legal risks that innovation brings. Therefore, after introducing the technical principles of IoV, it is essential to examine its legality and compliance to ensure that it can address previous technological loopholes in both technical and legal compliance aspects, thereby achieving significant breakthroughs. Through a legal analysis of the IoV system, it is evident that the IoV system relies on a hybrid contract framework, comprising both traditional contracts and smart contracts. Such a framework can achieve the legal objectives stipulated by data protection laws and digital regulations across different jurisdictions, such as the EU General Data Protection Regulation (GDPR) and China's Personal Information Protection Law (PIPL). This framework involves four types of entities (RSUs, vehicles, users, and insurance companies) and five types of contractual relationships (vehicle-to-RSUs, vehicle-to-vehicle, user-to-vehicle, user-to-user, and insurance-to-user). By adopting this contract-based approach, the IoV system enhances the legal compliance requirements for autonomous driving, particularly with significant improvements in privacy protection and data security. Specifically, the IoV system should be governed by a combination of smart contracts and traditional contracts. While smart contracts offer numerous advantages for signatories and play a significant role in the IoV, the combination of smart contracts and traditional contracts could better achieve the legal values of the IoV system.

Contributions

- This paper extends BeACONS [28] by proposing and evaluating a blockchain-enabled authentication and communication network for scalable IoV. It utilizes BeMutual [29], a blockchain-based decentralized identity management and end-to-end mutual authentication communication protocol, enhancing communication security in IoV and eliminating the reliance on centralized infrastructure. Additionally, it incorporates BeDNS [30], a blockchain-based domain name system that provides domain binding and query for decentralized identities in BeMutual, improving system usability and semantic capabilities.
- In this paper, tests are conducted on three typical scenarios in BeACONS, including (1) an unauthorized RSU entering the communication range, (2) an authorized RSU entering the communication range, and (3) an established RSU leaving the communication range. Transactions per second (TPS) and latency for these scenarios were collected and analyzed.
- This paper defines the four types of participants and five types of contractual relationships in a decentralized, contract-based IoV system, validating that smart contracts hold the same legal validity as traditional contracts and can ensure legal compliance across different jurisdictions. The IoV contract system relies on the principle of autonomy of will, making it less susceptible to varying legal jurisdictions. Additionally, this paper is framed within the context of the GDPR, which serves as a representative framework for data law across multiple jurisdictions.
- This paper also identifies the shortcomings of smart contracts within the IoV system, primarily their inability to balance interests between equal rights. This limitation arises from the inherently restricted content of smart contracts, their lack of interpretability, and the difficulty in modifying them. These deficiencies necessitate the supplementation of traditional contracts to address and resolve such issues.

This paper is organized as follows: Section 2 reviews advancements and gaps in certificateless protocols and blockchain applications for the Internet of Vehicles. Section 3 explores the system architecture, detailing primary and sub-layer components. Section 4 provides the implementation details, including smart contract deployment and dynamic RSU management. Section 5 discusses experimental results that validate the effectiveness of the system. Section 6 examines the legal implications of integrating smart and traditional contracts in the IoV context. This paper concludes in Section 7 with a summary of findings and suggestions for future research.

2. Related Works

This section reviews the existing literature on certificateless protocols and decentralized schemes based on blockchain for IoV, identifying gaps addressed by this paper.

2.1. Certificateless Protocols

Cui et al. [18] introduced a certificateless aggregate signature (CLAS) scheme for V2I communications that did not require pairings. However, Kamil and Ogundoyin [31] demonstrated that the CLAS scheme proposed in [18] is vulnerable to attacks by a Type II adversary $\mathcal{A}_2$, which can be executed in polynomial time. As a response, the authors of [31] proposed an improved CLAS scheme based on elliptic curve cryptography (ECC) specifically tailored for vehicular ad hoc networks. Despite this refinement, the scheme still exhibits weaknesses highlighted in [32], where it remains susceptible to attacks by both the Type I adversary $\mathcal{A}_1$ and the Type II adversary $\mathcal{A}_2$. Presently, researchers such as Xie et al. [33] and Genc et al. [34] have shifted their focus beyond certificateless schemes to delve into the realm of conditional privacy preservation. In this context, Xu et al. [29] introduced the blockchain-enabled mutual authentication protocol (BeMutual) as an innovative approach to secure and privacy-preserving peer-to-peer communications. Experimental results indicate that BeMutual offers improvements in communication efficiency and computational overhead compared to established communication authentication protocols like TLS 1.3 and IKEv2. Besides Cui et al., Mei et al. [35] also proposed an efficient certificateless aggregate signature with conditional privacy preservation in IoV. Considering the potential lack of well-established infrastructures in real-world scenarios, Tan et al. [36] proposed an efficient UAV certificateless group authentication mechanism to facilitate secure data transmission in infrastructure-less IoVs.

2.2. Decentralized Scheme Based on Blockchain for IoV

Blockchain technology has found applications across various domains, including radio access network (RAN) [37,38], Internet of Things (IoT) [39], blockchain-enabled resource management and sharing for 6G communications [40], a privacy-preserving blockchain platform for a data marketplace [41], and federated learning (FL) [42], heralding the advent of the Web3 era [43]. In the context of Web3, Zhou et al. [30] implemented blockchain-enabled domain name service (BeDNS) as a decentralized name system for BeMutual. BeDNS simplifies complex blockchain addresses into user-friendly domain names and associates blockchain addresses with the network interface identifiers of their respective owners, facilitating secure verification during BeMutual connections. In the realm of IoV, Jabbar et al. [44] proposed the blockchain-based decentralized IoT solution for vehicle communications (DISV), designed to address security, centralization, and privacy leakage concerns in V2X communications. For software-defined networks (SDNs) in IoV, Vishwakarma et al. [45] introduced a lightweight blockchain-based security protocol known as lightweight blockchain-based security protocol for secure communications and storage in SDN-enabled IoV (LBSV). LBSV operates as a permissioned blockchain network, utilizing a modified practical Byzantine fault tolerance (mPBFT) consensus algorithm proposed by the authors.

Similar scenarios to IoV have been tested across various domains, such as the novel authentication scheme for UAV–ground station and UAV-UAV communication (SecAuthUAV) [46], which can be accelerated by programmable switches [47]. This paper extends the scope of BeACONS [28] by implementing the system and conducting simulations. Several relevant scenarios are tested, followed by a discussion on potential sociological issues within the BeACONS model.

3. System Design

This paper adopts BeMutual as the identity management system and encrypted communications protocol for IoV interconnections. It also adopts BeDNS as the decentralized name system for inter-vehicle communications and intra-vehicle communications. The proposed communications system thus comprises a primary layer for inter-vehicle communications and a sub-layer dedicated to intra-vehicle communications for each vehicle.

3.1. The Structure of the Primary Layer

The primary layer as shown in Figure 1 facilitates the establishment of inter-vehicle connections, integrating BeMutual into BeDNS to ensure secure communications. Key elements of this layer include its entities and their attributes.

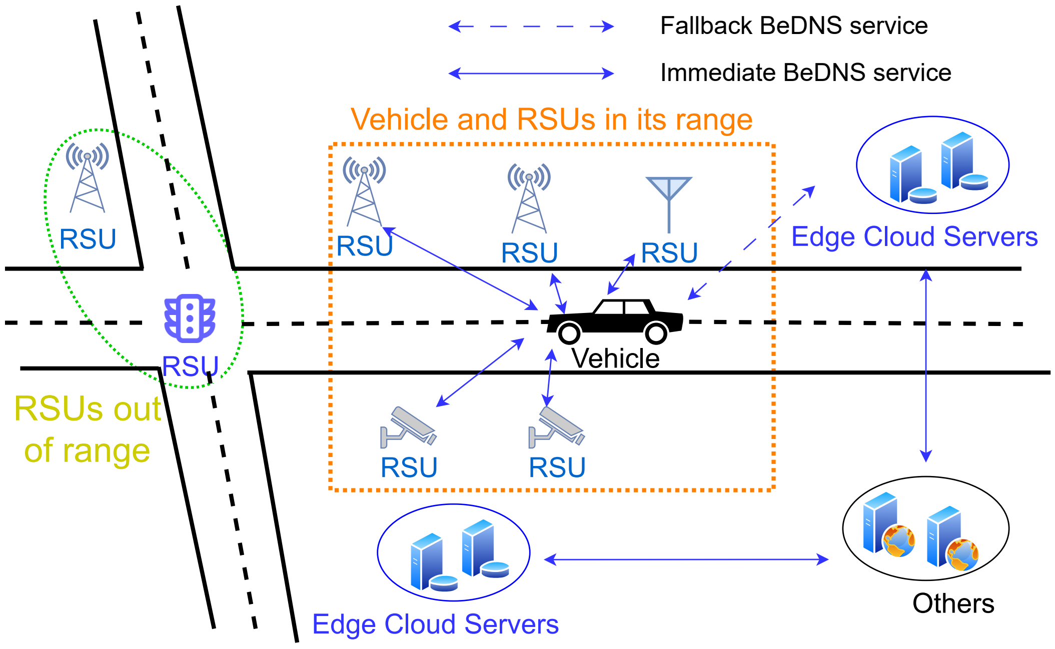

Figure 1. Overall structure of the primary layer.

3.1.1. Entities in the Primary Layer

These include all devices engaged in inter-vehicle communications, each identified by a unique blockchain address (BCADD). As shown in Figure 1, entities are categorized into three groups:

- **RSUs and edge servers:** They provide BeDNS services and maintain the BeDNS blockchain. Their responsibilities include managing blockchain identity through Bind, Update, Verify, and Search functions, and mapping among BCADDs, corresponding network locations, and network interface identifiers. RSUs serve nearby vehicles immediately under normal conditions, while edge servers act as fallbacks for vehicles and provide immediate service to the entities in the primary layer that do not require RSU-based BeDNS service.
- **Clients:** Clients are typically vehicles, which use BeMutual to establish secure connections and interact with BeDNS servers for creating, updating, verifying, and searching mapping information.
- **Cloud servers:** These provide various cloud services, including computing assistance for autonomous driving, over-the-air (OTA) updates, and remote diagnostics.

The three groups above together constitute the primary layer, and are crucial participants in inter-vehicle communications.

3.1.2. Attributes for Entities in the Primary Layer

To optimize system performance and protect RSUs from cyberattacks, entities in this layer exhibit four specific attributes.

- **BeMutual/BeDNS priorities:** To guarantee the lightweight and specialized nature of the system while also mitigating compatibility concerns, the utilization of BeMutual and BeDNS is limited to activities associated with safe driving and the handling of privacy-sensitive information. The determination of whether BeDNS or/and BeMutual are employed in a connection is contingent upon the pre-determined level of criticality, as specified by established protocols such as the routing table.
- **Partial isolation of RSUs:** Within this model, RSUs are presumed to be inter-connected via the Internet. To safeguard against potential distributed denial of service (DDoS) attacks originating from the Internet and to ensure the exclusive dedication of these RSUs to IoV, they are configured to solely entertain service requests originating from

direct wireless connections established with nearby vehicles. This strategy can be readily and reliably implemented due to the typical physical separation of such links and RSU-to-Internet connections through distinct network interfaces.

- **Limited service range of RSUs:** RSUs catering to nearby vehicles naturally exhibit a confined service range, owing to the restricted coverage of wireless signals facilitating vehicle-to-RSU communications. As depicted in Figure 1, the availability of RSUs for an individual vehicle fluctuates continuously as the vehicle traverses in and out of the coverage areas of RSUs. During an RSU's available time slot (ATS), it extends BeDNS services to the associated vehicle.
- **Access to Secret Keys (SKs):** SKs, from which BCADDs are derived, should be retained by the entity owners represented by the BCADDs.

The four attributes above can ensure optimal system performance and safeguard RSUs against cyber threats, which will be reflected in the later simulations and results section.

3.2. The Structure of the Sub-Layer

The sub-layer as shown in Figure 2 handles intra-vehicle connections via BeMutual without utilizing BeDNS, with each vehicle possessing a unique sub-layer. Key elements include units and their roles.

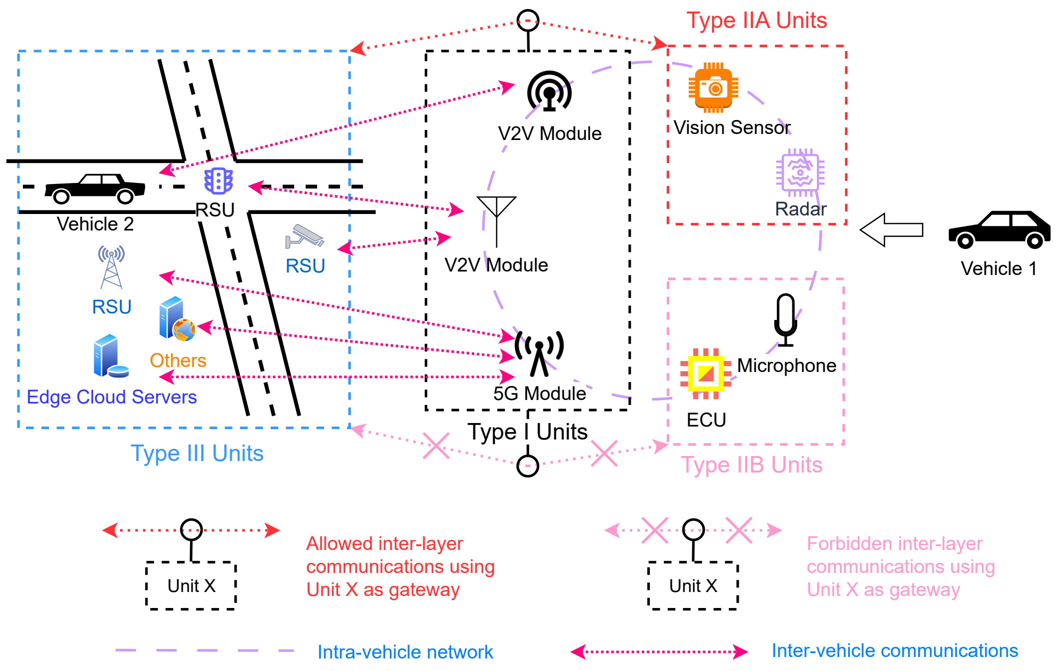

Figure 2. Overall structure of the sub-layer.

3.2.1. Units in the Sub-Layer

Units are categorized into four types based on their communication capabilities, each of which has its own unique BCADD.

- **Type I** units are capable of both inter- and intra-vehicle communications. Each Type I unit possesses a unique BCADD recorded in the BeDNS blockchain rather than sharing the vehicle's.
- **Type IIA** units are limited to intra-vehicle communications but can participate in inter-vehicle communications via Type I units, with their BCADDs also recorded in the BeDNS blockchain.
- **Type IIB** units are restricted to intra-vehicle communications only, with no BeDNS mapping due to privacy and safety concerns.

- **Type III** units are typically external to a particular intra-vehicle network but capable of connecting Type I and Type IIA units through inter-layer interactions, with BCADDs recorded in the BeDNS blockchain.

The four types of units above are interconnected according to their respective communication protocols, collectively forming the sub-layer.

3.2.2. Functionalities of Units in the Sub-Layer

To maintain privacy, integrity, and security in a lightweight manner, the sub-layer has two primary functionalities:

- **Communications identity management:** Type I units manage mapping information between BCADDs of the four types of units and their network identifiers within the intra-vehicle network.
- **Conditional mapping:** Units decide independently whether to upload their mapping information to the BeDNS blockchain based on their exclusive possession of the correct SK to create valid signatures for BeDNS functions.

The two major roles above ensure the best performance of intra-vehicle communications, making contributions to preventing privacy leaks and defending against cyber attacks.

3.3. Interactions in the Proposed System

Entities in the primary layer and units in the sub-layer interact to realize secure BeMutual connections. The interactions can be divided into three types: (1) interactions within the primary layer, (2) interactions within the sub-layer, and (3) inter-layer interactions.

- **Interactions within the primary layer** refer to interactions in the primary layer involving RSUs, edge servers, vehicles, and cloud servers as shown in Figure 1. RSUs and edge servers provide BeDNS service to vehicles within the service range, while cloud servers provide cloud services to both of them. To be more specific regarding vehicles in reality, the Type I units of the vehicles, rather than a general model of indivisible vehicles, engage in interactions within the primary layer.
- **Interactions within the sub-layer** refer to the interactions occurring solely within the sub-layer of a single vehicle, involving only Type I, Type IIA, and Type IIB units without the involvement of BeDNS.
- **Inter-layer interactions** refer to interactions that combine the two types of previously mentioned interactions, involving RSUs, edge servers, vehicles, cloud servers, and Type I/IIA/III units. The proposed interactions can be categorized into three types specifically: (1) vehicle-to-RSU, (2) vehicle-to-server, and (3) vehicle-to-vehicle.
    1. **Vehicle-to-RSU**
       The Type IIA unit on the vehicle V drives the corresponding Type I unit on V to interact with an RSU nearly as shown in Figure 3. It features direct communication between the Type I unit and the RSU without involving any intermediary.
    2. **Vehicle-to-server**
       The Type I unit on V is driven by the Type IIA unit to establish interaction with servers, including the edge servers and the cloud servers as shown in Figure 3. This interaction differs from (1) in that, due to the inability to establish direct communication with the server, the Type I unit must interact with the servers through intermediaries such as nearby 5G base stations.
    3. **Vehicle-to-vehicle**
       The Type IIA units on two different vehicles establish interaction with the BeDNS service provided by RSUs and/or the edge servers as shown in Figure 4. This is the most complex interaction in the system, which will be described in detail in the implementation section.

The interactions above guarantee the security of the BeMutual connections with the BeDNS services, enhancing the privacy of the system.

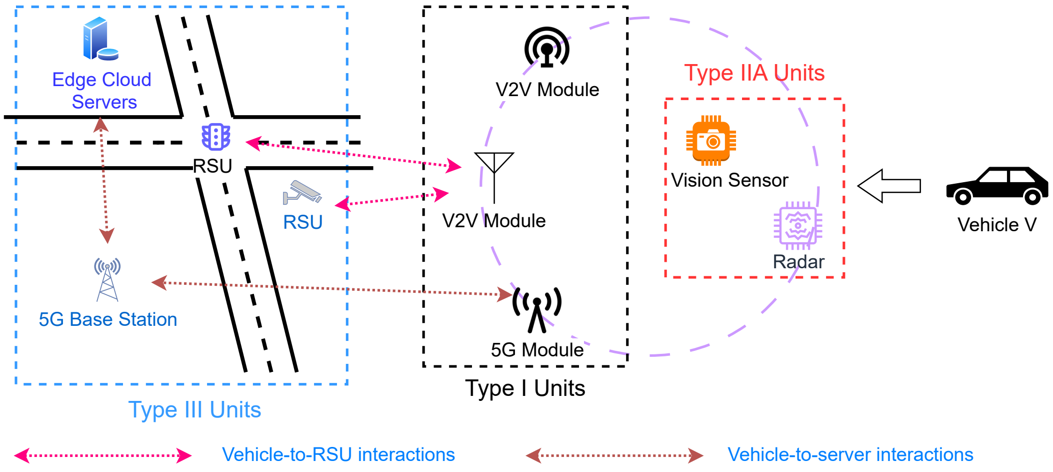

Figure 3. Two types of interactions: vehicle-to-RSU and vehicle-to-server.

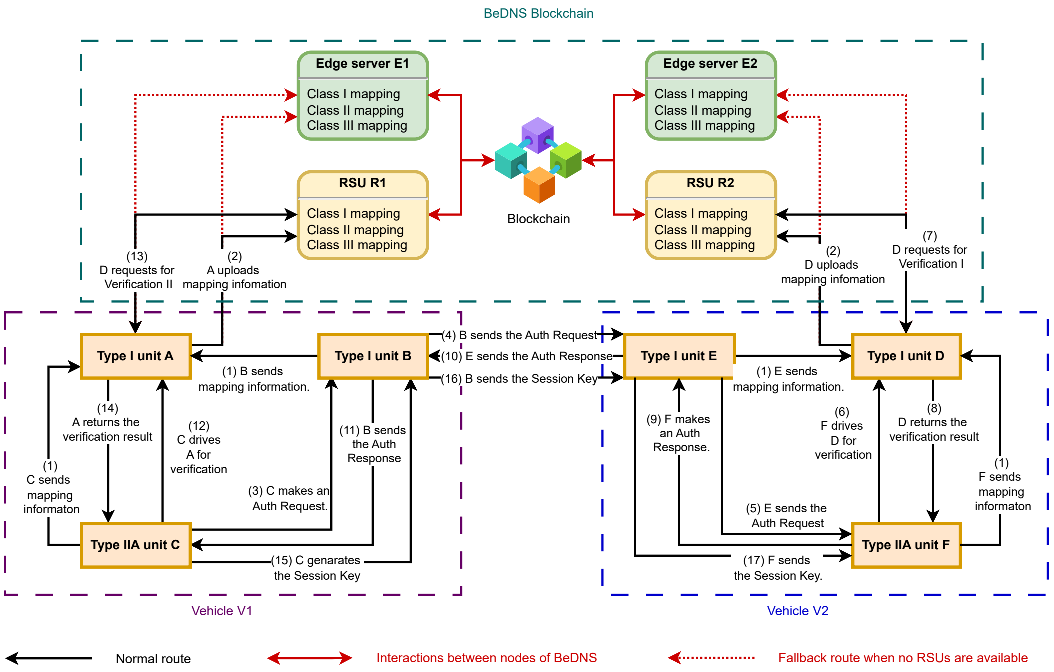
BeDNS Blockchain
Edge server E1
Class I mapping
Class II mapping
Class III mapping
Edge server E2
Class I mapping
Class II mapping
Class III mapping
RSU R1
Class I mapping
Class II mapping
Class III mapping
RSU R2
Class I mapping
Class II mapping
Class III mapping
Blockchain
(13) D requests for Verification II
(2) A uploads mapping infomation
(2) D uploads mapping infomation
(7) D requests for Verification I
Type I unit A
Type I unit B
Type I unit E
Type I unit D
(4) B sends the Auth Request
(10) E sends the Auth Response
(16) B sends the Session Key
(1) B sends mapping information.
(1) E sends mapping information.
(14) A returns the verification result
(12) C drives A for verification
(11) B sends the Auth Response
(1) C sends mapping informaton
(3) C makes an Auth Request.
(15) C genarates the Session Key
(9) F makes an Auth Response.
(6) F drives D for verification
(8) D returns the verification result
(1) F sends mapping informaton
(5) E sends the Auth Request
(17) F sends the Session Key.
Type IIA unit C
Type IIA unit F
Vehicle V1
Vehicle V2
Normal route
Interactions between nodes of BeDNS
Fallback route when no RSUs are available

Figure 4. The inter-layer BeMutual session between two Type IIA units on two vehicles separately.

3.4. Smart Contract Design

The smart contract, integral to the IoV framework, is deployed on the blockchain to efficiently manage and secure data interactions between vehicles and infrastructure components. This design ensures that all transactions within the IoV ecosystem are handled securely, with a strong emphasis on maintaining data integrity and reliability across the network. By leveraging the inherent properties of blockchain technology, such as immutability and transparency, the smart contract significantly enhances system security and operational efficiency. This architecture mitigates various security threats and provides the scalability necessary to adapt to evolving IoV demands.

3.4.1. Data Structures

The smart contract is comprised of three primary data classes, each optimized for different components within the IoV ecosystem:

- The first class handles basic infrastructure data, focusing on elements crucial for the operation and management of network components. This includes identifiers for unique entity recognition and network parameters critical for establishing and maintaining communication.

- The second class is dedicated to vehicle-specific data, incorporating identifiers and status information that allow for the tracking and management of vehicle interactions within the network. This class also includes interfaces for user interaction, enhancing the user experience by providing clear and accessible vehicle data.
- The third class extends the capabilities of the second by introducing additional layers of data complexity, which accommodate advanced IoV functionalities such as multi-device interactions and secondary user interface integrations.

3.4.2. Functional Capabilities

The smart contract offers several critical functionalities:

- **Upload**: This functionality allows for the initial recording of data into the blockchain. It is designed to handle various types of data inputs from different IoV components, ensuring that all information is securely logged and timestamped.
- **Update**: Essential for maintaining the relevance and accuracy of the stored data, this functionality supports the modification of existing records. It includes safeguards to verify the legitimacy of the modification requests, ensuring that only authorized changes are made.
- **Verification**: To maintain the integrity of the IoV system, this functionality is crucial for ensuring the authenticity and trustworthiness of communication between vehicles and infrastructure components. It achieves this by comparing externally provided identity data with the verified data stored on the blockchain. This verification process ensures that all interactions within the network are conducted between authenticated and authorized entities, thereby safeguarding the system against identity spoofing and ensuring that all communications are secure and trustworthy.

Each functionality as shown in Figure 5 is designed to ensure that all data transactions within the smart contract are secure, reliable, and consistent with the high standards required for blockchain-based applications in vehicular environments. This strategic arrangement of data handling and processing capabilities supports the overarching goals of the IoV system, facilitating efficient, secure, and scalable interactions across the network.

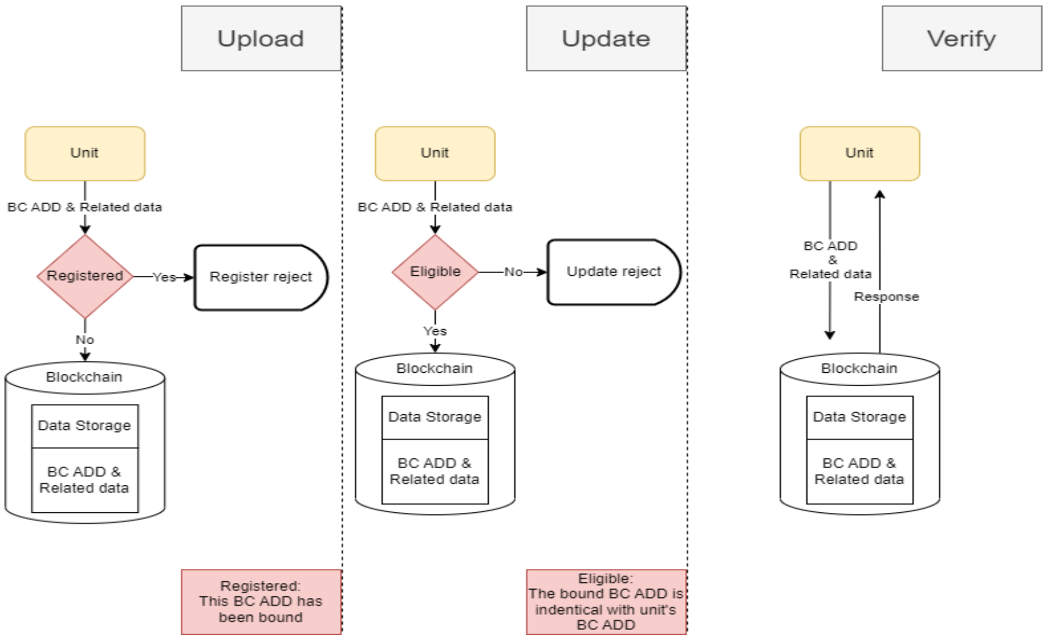

Figure 5. Smart contract design.

4. Implementation

The implementation leverages an Intel i7-13620H CPU at 2.40 GHz with a 16 GB RAM computer. The smart contract is written in Solidity and deployed on the Ethereum blockchain, utilizing three nodes. The dynamic update of RSUs and IoV communication are implemented in Python.

4.1. Smart Contract Implementation

The smart contract for the IoV is implemented on the Ethereum blockchain using Solidity. This implementation efficiently manages complex interactions between vehicles and infrastructure, ensuring robust data integrity and security.

4.1.1. Data Structures

The smart contract organizes data into three primary structured classes tailored for distinct aspects of the IoV network, as detailed in Table 1. Each class is optimized to manage specific data interactions efficiently, ensuring data integrity and supporting secure communications across the network. Class I handles server-related information, Class II focuses on primary vehicle operations, and Class III incorporates secondary user interface devices.

Table 1. Data structure in the smart contract.

Class	**Element**
Class I Unit	BCADDServer ADDServer Timestamp Signature
Class II Unit	BCADDCar BCADDUI ADDUI LabelUI Timestamp Signature
Class III Unit	BCADDUIIa ADDUIIa LabelUIIa Timestamp Signature

- **Class I Unit:** The Class I unit primarily manages server-related information, essential for the network's operations. It stores the *BCADDServer*, enabling unique identification within the blockchain network. Additionally, the *ADDServer* is recorded to facilitate connectivity and communications. Each entry in this class automatically captures a *timestamp* at the time of data entry, ensuring that the timing of data storage is recorded for future reference. To guarantee the authenticity of the data, a cryptographic *signature* is also included with each record.
- **Class II Unit:** Focused on vehicle-specific information, this class caters to the core interactions within the IoV ecosystem. It captures the *BCADDCar* and its user interface unit *BCADDUI*, thus identifying both the vehicle and its associated control interfaces. The *ADDUI* is stored to maintain communication channels, and a descriptive *labelUI* provides additional information about the interface's functionality or role. Similar to Class I, this class also maintains a *timestamp* and a cryptographic *signature* with each entry to validate the timing and integrity of the data stored.
- **Class III Unit:** Building upon the foundation set by Class II, the Class III Unit introduces additional layers of data to handle complex scenarios involving secondary user interface devices. This includes storing the *BCADDUIIa* and the *ADDUIIa* of secondary devices, which may provide enhanced functionalities or additional control

options within the vehicle ecosystem. Each secondary device is also described using a *labelUIIa* that gives further details about its purpose and use. As with the other classes, entries in Class III are *timestamped* and signed cryptographically, ensuring that data entries are both time-stamped and secured.

4.1.2. Mappings

Each class is connected to the blockchain through specialized mappings that ensure efficient data management and retrieval. These mappings, as shown in Table 2, facilitate organized data storage and accessibility by structuring the association of data with blockchain addresses. Below are the roles and functions of each mapping:

- **Class I Mapping:** Establishes a direct connection between blockchain addresses and instances of Class I units, which manage server-related information. This mapping ensures that server data can be quickly and accurately retrieved from the blockchain without redundancy.
- **Class II Mapping:** Designed to handle complex associations, this mapping concatenates the blockchain addresses of vehicles (*bcaddCar*) and their user interfaces (*bcaddUI*) to form a unique key for each Class II unit instance. It is crucial for organizing and retrieving vehicle-specific data, thus enhancing operational efficiency.
- **Class III Mapping:** Incorporates a more intricate structure by combining blockchain addresses of vehicles, primary user interfaces, and secondary user interfaces (*bcaddUIIa*). This allows for the management of multi-layered data structures that are typical in complex IoV systems, ensuring that all related data components are accessible through a single compound key.
- **Label to Class II Mapping:** Utilizes labels to categorize and facilitate quick access to Class II units, streamlining interactions and improving the speed of data retrieval for vehicle-related operations.
- **Label to Class III Mapping:** This mapping organizes Class III units by their labels, significantly simplifying the navigation and extraction of data regarding secondary devices, which are often critical in advanced IoV applications.

Table 2. Mapping structures in the smart contract.

Mapping	Key	Value
Class I	BCADDServer	Class I Unit Instance
Class II	Concatenated BCADDCar and BCADDUI	Class II Unit Instance
Class III	Concatenated BCADDCar, BCADDUI, and BCADDUIIa	Class III Unit Instance
Label to Class II	Label	List of Class II Unit Instances
Label to Class III	Label	List of Class III Unit Instances

4.1.3. Functionalities

The smart contract incorporates a series of functions that are essential for managing data within the blockchain environment. These functions, as shown in Table 3, are structured to ensure the integrity and systematic management of data relevant to the IoV operations.

- **Upload Functions**:
  - *UploadI:* This function inputs server-specific data into the blockchain, accepting parameters such as the blockchain and network addresses of the server, along with a digital signature. It automatically records the transaction's timestamp using **block.timestamp** to chronologically tag each entry.
  - *UploadII:* Designed for vehicle data, this function logs the blockchain addresses of both the vehicle and its user interface. It also captures the network address,

a descriptive label, a digital signature, and a timestamp, forming a unique key by concatenating the vehicle's and UI's blockchain addresses for data indexing.

  – *UploadIII:* An extension of UploadII, this function includes parameters for additional user interfaces, thus accommodating more complex data relationships within the IoV ecosystem.

- **Update Functions**:
  – *UpdateI:* Updates server data on the blockchain by first verifying the existence of the server's blockchain address, thereby preventing unauthorized modifications.
  – *UpdateII:* This function updates vehicle data after confirming the existence of a composite key created from the vehicle's and UI's blockchain addresses.
  – *UpdateIII:* Targets data involving complex interactions among multiple devices by ensuring the existence of concatenated keys that include the vehicle, primary UI, and secondary UI blockchain addresses.
- **Verification Functions**:
  – *VerifyI:* This function ensures the integrity and authenticity of server data by comparing the hashed network address stored in the blockchain with the provided network address.
  – *VerifyII:* It verifies the data integrity of both the vehicle and its user interface by hashing and comparing essential data fields, including network addresses and labels.
  – *VerifyIII:* Specifically designed for systems with secondary interfaces, this function extends the capabilities of VerifyII by adding further verification processes for secondary UIs.

Table 3. Functions in the smart contract.

Function Name	Data Processed	Response
Upload I	BCADDServer, ADDServer, Signature, Timestamp	Upload Response
Upload II	BCADDCar, BCADDUI, ADDUI, LabelUI, Signature, Timestamp	Upload Response
Upload III	BCADDCar, BCADDUI, ADDUI, LabelUI, Signature, Timestamp, BCADDUiia, ADDUIIa, LabelUIia	Upload Response
Update I	BCADDServer, ADDServer, Signature, Timestamp	Update Response
Update II	BCADDCar, BCADDUI, ADDUI, LabelUI, Signature, Timestamp	Update Response
Update III	BCADDCar, BCADDUI, ADDUI, LabelUI, Signature, Timestamp, BCADDUiia, ADDUIIa, LabelUIIa	Update Response
Verify I	BCADDServer, ADDServer	Validation Response
Verify II	BCADDCar, BCADDUI, ADDUI, LabelUI	Validation Response
Verify III	BCADDCar, BCADDUI, ADDUI, BCADDUIIa, LabelUI, ADDUIIa, LabelUiia	Validation Response

Through these implementations, the smart contract ensures that all vehicle-to-infrastructure interactions within the IoV are recorded securely and immutably, enhancing both operational efficiency and data security.

4.2. Dynamic Feature of RSU Availability

As the vehicle travels along the road, it continuously moves in and out of the service areas of different RSUs. For a group of RSUs denoted as G that serves a vehicle denoted as V, they integrate new RSUs that come into range and exclude those that fall out of range, which is based on practical byzantine fault tolerance (PBFT) [48]. The RSUs in G are nodes and V is the client, with the primary node denoted as R_p, the new RSU denoted as R_n, and one common RSU in G denoted as R_i.

4.2.1. Joining of an RSU

Relying on the periodic broadcast of heartbeat messages from RSUs, vehicles can determine whether a particular RSU is communicable. As shown in Algorithm 1, the procedure for integrating an RSU will be demonstrated in detail later. The symbols used in the procedure are defined as shown in Table 4.

Algorithm 1 Joining of new RSU

```
1: procedure INTEGRATION
2:     if $V$ receives heartbeat messages of a new RSU $R_n$ then
3:         $V$ start BeMutual Authentication with $R_n$
4:         if BeMutual session is established then
5:             $V$ sends $Req_{Int}(R_n)$ to $R_p$
6:             if PBFT consensus in $G$ on $Req_{Int}(R_n)$ is reached then
7:                 $R_i$ performs Int($R_n$)
8:                 $N_{Ri} \leftarrow N_{Ri} + 1$
9:                 $R_i$ sends $Rpl_{Int}(R_n)$ to $V$
10:                if reception of $\left(\frac{N_v+2}{3}\right)$ $Rpl_{Int}(R_n)$ at $V$ then
11:                    $V$ executes $Cnf_{Int}(R_n)$
12:                    $N_V \leftarrow N_V + 1$
13:                end if
14:            end if
15:        end if
16:    end if
17: end procedure
```

First of all, when V receives heartbeat messages of a new RSU R_n, V will start the BeMutual authentication with R_n. After the BeMutual session is established, $Req_{Int}(R_n)$ will be sent from V to R_p. Then, for every R_i, if PBFT consensus in G on $Req_{Int}(R_n)$ is reached, R_i will perform Int(R_n), making an increment of 1 to N_{Ri}, and finally the $Rpl_{Int}(R_n)$ will be sent to V. When the number of $Rpl_Int(R_n)$ reaches $(\frac{N_v+2}{3})$, $Cnf_{Int}(R_n)$ will be performed, with N_v incremented by 1, after which the joining of R_n is finished. No view change occurs in the procedure since R_p remains the same.

Table 4. Symbols used in the algorithm and their meanings.

Symbol	Meaning
V	The vehicle in the join/exclusion process
$Req_{Int}(R_n)$	The request for joining of R_n
$Req_{Exc}(R_n)$	The request for exclusion of R_n
G	The group of RSUs serving V
R_p	The primary node RSU in G
R_i	One common node RSU in G before view change
R_i'	One common node RSU in G after view change
N_{Ri}	The number of RSUs in G recorded by R_i
$N_{Ri'}$	The number of RSUs in G recorded by R_i'
N_V N_V	The number of RSUs in G recorded by V The number of RSUs in G recorded by V
R_n	The new RSU to be integrated or the old RSU to be excluded

Table 4. *Cont.*

Symbol	Meaning
Int(R_n)	The procedure that R_i integrates R_n into G recorded by R_i
Exc(R_n)	The procedure that R_i excludes R_n out of G recorded by R_i
$Rpl_{Int}(R_n)$	The reply from R_i regarding the completion of Int(R_n)
$Rpl_{Exc}(R_n)$	The reply from R_i regarding the completion of Exc(R_n)
$Cnf_{Int}(R_n)$	The procedure where V confirms the joining of R_n
$Cnf_{Exc}(R_n)$	The procedure where V confirms the exclusion of R_n
VC(R_p)	The view change that prevents R_p from voting and being elected as the new primary node

4.2.2. Exclusion of an RSU

The procedure for excluding a new RSU is shown in Algorithm 2, which is slightly different from the join procedure. The symbols in this procedure are also defined as shown in Table 4. The view change can occur in the procedure as the RSU to be excluded may be R_p.

Firstly, when V loses heartbeat messages of R_n, V will broadcast $Req_{Exc}(R_n)$ to every R_i. After that, if PBFT consensus in G on $Req_{Exc}(R_n)$ is reached, two different procedures will be performed based on whether R_n is R_p. If yes, VC(R_p) will be performed, after which R_i' will perform Exc(R_n), making a decrement of 1 to $N_{Ri'}$, and finally the $Rpl_{Exc}(R_n)$ will be sent to V. In contrast, if not, the view will stay the same, and R_i will perform Exc(R_n), make a decrement of 1 to N_{Ri}, and finally send the $Rpl_{Exc}(R_n)$ to V. When the number of $Rpl_Exc(R_n)$ reaches ($\frac{N_v+1}{3}$), $Cnf_{Exc}(R_n)$ will be performed, with N_v decremented by 1. Then, the exclusion of R_n is completed.

The dynamic feature of RSU availability ensures that the RSUs in G remain within a communicable distance of V, thereby facilitating the system's timeliness and efficiency.

Algorithm 2 Exclusion of an RSU out of the service area

```
procedure EXCLUSION
    if $V$ loses heartbeat messages of RSU $R_n$ then
        $V$ broadcasts $Req_{Exc}(R_n)$ to $R_i$
        if PBFT consensus in G on $Req_{Exc}(R_n)$ is reached then
            if $R_n$ is $R_p$ then
                $VC(R_p)$ is performed
                $R_i'$ performs $Exc(R_n)$
                $N_{Ri'} \leftarrow N_{Ri'} - 1$
                $R_i'$ sends $Rpl_{Exc}(R_n)$ to $V$
            else
                $R_i$ performs $Exc(R_n)$
                $N_{Ri} \leftarrow N_{Ri} - 1$
                $R_i$ sends $Rpl_{Exc}(R_n)$ to $V$
            end if
            if reception of $\left(\frac{N_v+1}{3}\right)$ $Rpl_{Exc}(R_n)$ at $V$ then
                $V$ performs $Cnf_{Exc}(R_n)$
                $N_V \leftarrow N_V - 1$
            end if
        end if
    end if
end procedure
```

4.3. BeMutual Connections

Entities and units engage with each other to form secure BeMutual connections. The most general scenario where an inter-layer BeMutual session between two Type IIA units on two vehicles separately is established will be described in detail as shown in Figure 4, and can be simplified to other kinds of scenarios. The symbols used in this part are defined as shown in Table 5, while the detailed contents of the transmitted information are provided in Table 6,

Table 5. Symbols used in the inter-layer BeMutual session.

Symbol	**Meaning**
V_1/V_2	The vehicles in the inter-layer BeMutual session
A/D	The Type I units on V_1/V_2 specialized in communicating with RSUs and edge servers
B/E	The Type I units on V_1/V_2 specialized in communicating with Type I units in other vehicles
C/F	The Type IIA units on V_1/V_2
$BCADD_{V1}/$ $BCADD_{V2}$	The blockchain address of V_1/V_2
$ADD_{V1}/$ ADD_{V2}	The address of V_1/V_2
PK_{V1}/PK_{V2}	The public key of V_1/V_2
SK_{V1}/SK_{V2}	The private key of V_1/V_2
$nonce1/$ $nonce2$	A random value generated by V_1/V_2 and introduced to prevent attacks
$KeyMaterial$	The seed V_1 used to generate the session key
$(Content)_{Key}$	The $Content$ encrypted by the Key

Table 6. The contents of the transmitted information in the inter-layer BeMutual session.

Transmitted Info.	**Contents**
Auth Request	$BCADD_{V2}$, ADD_{V1}, PK_{V1}, $nonce1$, $(ADD_{V1}, nonce1)SK_{V1}$
Verification I	$BCADD_{V1}$, ADD_{V1}
Auth Response	$BCADD_{V1}$, ADD_{V1}, ADD_{V2}, PK_{V2}, $nonce2$, $(ADD_{V1}, ADD_{V2}, nonce2)SK_{V2}$
Verification II	$BCADD_{V2}$, ADD_{V2}
Session Key	$BCADD_{V2}$, ADD_{V2}, ADD_{V1}, PK_{V2}, $nonce2$, $(KeyMaterial)PK_{V2}$

- In steps 1 and 2, the mapping information between the BCADDs, ADDs, and labels of all the required Type I units and Type IIA units is uploaded to the BeDNS blockchain. Taking vehicle $V1$ for example, unit A is a Type I unit specialized in communicating with RSUs and edge servers, while Type I unit B is specifically responsible for communicating with Type I units in other vehicles. Units B and C must send their mapping information to A and sign it before uploading it to the BeDNS blockchain since only A can interact with RSUs and edge servers. Then, the mapping information of A, B, and C will be uploaded to the BeDNS blockchain via the smart contract before which A will sign it. The process above for units D, E, and F occurs similarly in vehicle V_2.
- In steps 3–5, an authentication request is transmitted from vehicle V_1 to vehicle V_2 to establish secure communication. The authentication request is initiated by C,

transmitted to E via B, and finally sent to F. The public key (PK) of V_1 is transmitted to V_2 in the authentication request in plain text, which is a crucial credential for V_2 to subsequently verify the identity of V_1 with BeDNS [29].

- In steps 6–8, V_2 authenticates the identity information of V_1. V_2 first checks whether the PK_{V1} in the authentication request can derive to $BCADD_{V1}$ [29]. Then, V_2 utilizes the PK_{V1} to verify the signature, which is signed by SK_{V1} [29]. After F finishes the procedures above, F utilizes D to access the smart contract for verifying the mapping information between $BCADD_{V1}$ and ADD_{V1}, with BeDNS based on PBFT.
- In steps 9–11, V_2 sends an authentication response to V_1. The authentication response is initiated by F and transmitted to C via E and B. Similar to steps 3–5, PK_{V2} is transmitted to V_1 in the authentication response in plain text, with which V_1 can verify the identity of V_2.
- In steps 12–14, V_1 authenticates the identity information of V_2. First, V_1 needs to ensure that the response comes from V_2 by verifying PK_{V2} and $BCADD_{V2}$ [29]. Then, V_2 compares the plain text and the decrypted ADD_{V2} from the message encrypted by SK_{V2} in the authentication response to validate ADD_{V2} [29]. Consistent with steps 6–8, after finishing the above process, C authenticates the mapping information between $BCADD_{V2}$ and ADD_{V2} by accessing the smart contract on BeDNS via A.
- In steps 15–17, the session key generated by V_1 is sent to V_2. After verifying the identity of V_2, V_1 can choose a new session key or generate a session key utilizing $nonce_1$ and $nonce_2$ in the authentication response as the seed [29]. After the session key is generated, C will send the session key to F via B and E. Finally, the BeMutual authentication is established.

With all the steps above, the secure BeMutual connection between V_1 and V_2 is established, ensuring the privacy and confidentiality of their communication.

5. Simulations and Results

In Section 5, four simulations of typical scenarios in the proposed system are described in detail, including the PBFT process, the inter-layer BeMutual session, the dynamic updating of the RSU communication group, and the effective RSUs. All the simulations are conducted on the local computer, which features a 13th-generation Intel Core i7-13620H center process unit (CPU) with 16 GB random access memory (RAM), operating at 2.40 GHz, running on the Windows operating system. The Python program used in the simulation was version 3.12.3, and the imported Python libraries include Web3.py, json.py, random.py, time.py, statistics.py, math.py, threading.py, and Queue.py. The above setups and parameters are shown in Table 7.

Table 7. Setups and parameters for Section 5.

Setups and Parameters	Value
CPU	13th-generation Intel Core i7-13620H (2.40 GHz)
RAM	16 GB
Operating system	Windows
Python (json.py, random.py, statistics.py, time.py, math.py, threading.py, Queue.py)	3.12.3
Web3.py	6.19.0

5.1. The Runtime of the PBFT Process

As a preliminary simulation for subsequent simulations B and C, this simulation aims to test the average transaction time of the PBFT process and the corresponding TPS to simplify the simulation processes of simulations B and C. In simulations B and C, the PBFT process is omitted. The formula for calculating their total average transaction time is

$$t_{\text{total}} = t_{\text{PBFT}} + t_x \tag{1}$$

Here, t_{total} is the average transaction time of the complete process, t_{PBFT} is the average transaction time of the PBFT process, and t_x is the average transaction time of the BeMutual process (t_{BeMutual}) in simulation B, or the average transaction time of the entering process (t_{entering}) or leaving process (t_{leaving}) in simulation C.

In the simulations conducted in this section, communication delays are not considered; therefore, the average transaction time obtained in simulations A, B, and C represent the lower bounds.

5.1.1. Scenario Design

The PBFT process is divided into five steps: request, pre-prepare, prepare, commit, and reply [49]. This simulation sets up five RSUs and simulates a PBFT process initiated by a vehicle V to verify the identity of a vehicle. In this simulation, one RSU is first randomly selected as the primary node, followed by the random selection of zero or one malicious node, with the remaining nodes being normal.

- **Simulation preparation**
    1. Utilizing Geth [50] technology to deploy a block-chain locally and start all RSU nodes
    This simulation utilizes Geth technology for local blockchain deployment and node generation. After deployment, all the RSU nodes are started for the next steps.
    2. Creating blockchain accounts for all RSU nodes
    In the IoV system, the blockchain accounts and ADDs of RSU nodes are distributed by the system. In this simulation, the blockchain accounts of RSU nodes are created using Geth technology, while their ADDs are generated by subsequent backend code.
    3. Writing a Python program using the web3.py [51] package to simulate the PBFT process
    Thanks to web3.py, interactions with Ethereum in a Python program are realizable. Web3.py connects to Ethereum by configuring a provider (e.g., HTTP-Provider) to communicate with an Ethereum node via the JSON-RPC protocol, enabling interactions such as sending transactions and querying blockchain data [51]. Utilizing web3.py, this simulation crafted a Python script to simulate the PBFT process, whose block diagram is shown in Figure 6.
- **Simulation Measures**
  In this simulation, the entire PBFT process is then simulated, and the average transaction time of this process is recorded.

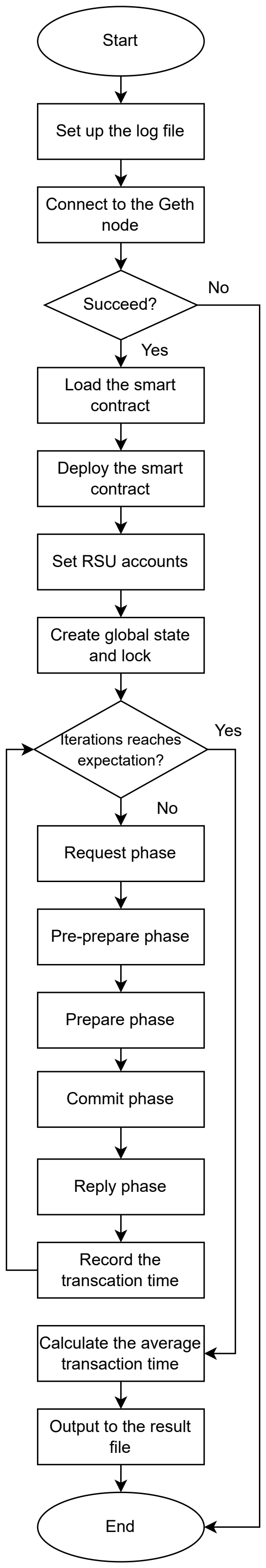
Start
Set up the log file
Connect to the Geth node
Succeed?
No
Yes
Load the smart contract
Deploy the smart contract
Set RSU accounts
Create global state and lock
Iterations reaches expectation?
Yes
No
Request phase
Pre-prepare phase
Prepare phase
Commit phase
Reply phase
Record the transcation time
Calculate the average transaction time
Output to the result file
End

Figure 6. The flow block diagram for the PBFT process.

5.1.2. Simulation Setups and Parameters

The basic setups and parameters are listed at the beginning of this section. The simulation employed Geth to establish five RSU nodes and facilitate communication among six units, involving two vehicles. In this simulation, the vehicle speed v is set to 1, the RSU service range radius r is 7, and the number of RSUs and vehicles in each scenario is 6 and 1, respectively.

5.1.3. Results

In this simulation, the average transaction time of the entire PBFT process is assessed across iterations ranging from 1000 to 10,000, with a step size of 1000. The simulation calculates the average of the obtained simulation results, yielding a result of 0.0393 s:

$$t_{\text{PBFT}} = 0.0393(s) \tag{2}$$

This simulation provides the constant parameter t_{PBFT} to simplify this process in subsequent simulations B and C, thereby establishing the lower bound for the average transaction time of the corresponding scenarios.

5.2. The Inter-Layer BeMutual Session

In this IoV system, ATS and overhead are two critically important performance parameters that receive significant attention. This simulation aims to simulate BeMutual communication, testing the overall runtime of the entire process as well as the time of a single transaction (upload, update, and verification), and thus calculating the transactions per second (TPS) accordingly.

5.2.1. Scenario Design

The simulation involves setting up five RSU nodes to simulate a BeMutual communication session between vehicle $V1$ and vehicle $V2$.

- **Simulation Preparation**
  1. Utilizing Geth technology to deploy a block-chain locally and start all RSU nodes
     This step is the same as the one in simulation A.
  2. Creating blockchain accounts for all RSU nodes
     This step is also the same as the one in simulation A.
  3. Creating a blockchain account for the vehicles and each component of the vehicles
     As a client, a vehicle's BCADD is derived from its PK and is therefore determined when the asymmetric key pair is generated during registration. This simulation simplifies this process, where blockchain accounts were created separately for vehicle 1, vehicle 2, and their respective units by the researchers via Geth directly, as the units on the vehicles have BCADDs independent of the vehicles. The above simplification will not affect the results of this simulation, as it is only part of the preparatory work and does not impact any steps within the simulation.
  4. Writing a Python program using the web3.py package to simulate the BeMutual session
     Utilizing web3.py, this simulation crafted a Python script, with a flow block diagram shown in Figure 7, to simulate the three core transactions and the BeMutual process.
- **Simulation Measures**
  1. The transaction time of upload, update, and verification
     This simulation tested the overhead of the three core transactions—upload, update, and verification—in the BeMutual process, and calculated the corresponding TPS. The purpose of the upload operation is to allow other vehicles to verify the mapping information of the vehicle and thus validate the identity of the vehicle, while the update operation allows clients to modify their mapping information as it changes.
  2. The runtime of the entire BeMutual process
     In this simulation, the entire BeMutual process duration was recorded, starting from the two vehicles initiating the upload of mapping information to the suc-

cessful establishment of the secure BeMutual connection. A secure BeMutual connection is a necessary prerequisite for communication between two vehicles in IoV systems, and hence the data from this simulation holds critical indicative significance for ATS expectations.

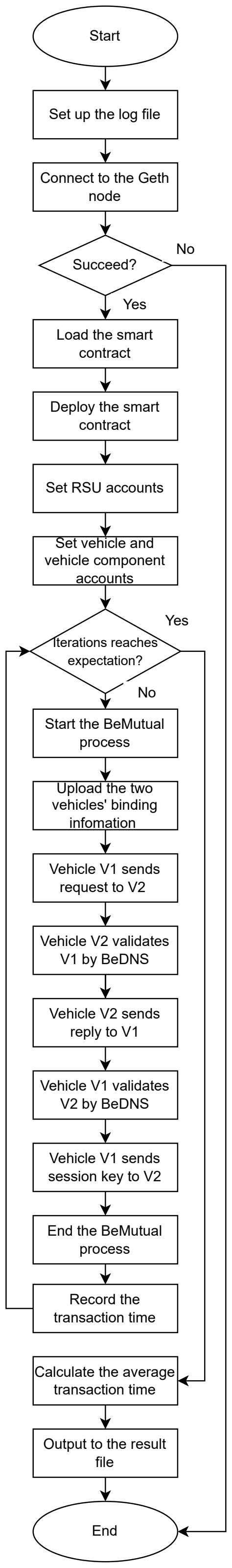
Start
Set up the log file
Connect to the Geth node
Succeed?
No
Yes
Load the smart contract
Deploy the smart contract
Set RSU accounts
Set vehicle and vehicle component accounts
Iterations reaches expectation?
Yes
No
Start the BeMutual process
Upload the two vehicles' binding infomation
Vehicle V1 sends request to V2
Vehicle V2 validates V1 by BeDNS
Vehicle V2 sends reply to V1
Vehicle V1 validates V2 by BeDNS
Vehicle V1 sends session key to V2
End the BeMutual process
Record the transaction time
Calculate the average transaction time
Output to the result file
End

Figure 7. The flow block diagram for the BeMutual process.

5.2.2. Simulation Setups and Parameters

The basic setups and parameters are listed at the beginning of this section. This simulation deployed five RSUs, two vehicles, and six units in total to simulate the BeMutual session.

5.2.3. Results

This simulation assessed the performance of the three types of transactions and the entire BeMutual session across iterations ranging from 1000 to 10,000, with a step size of 1000. The average transaction time and the corresponding TPS were computed for each transaction above. The results were then graphically depicted, with the average transaction time for the three single transactions depicted in one graph as shown in Figure 8, the TPS for them as shown in Figure 9, the average transaction time for the entire process as shown in Figure 10, and the TPS for it as shown in Figure 11.

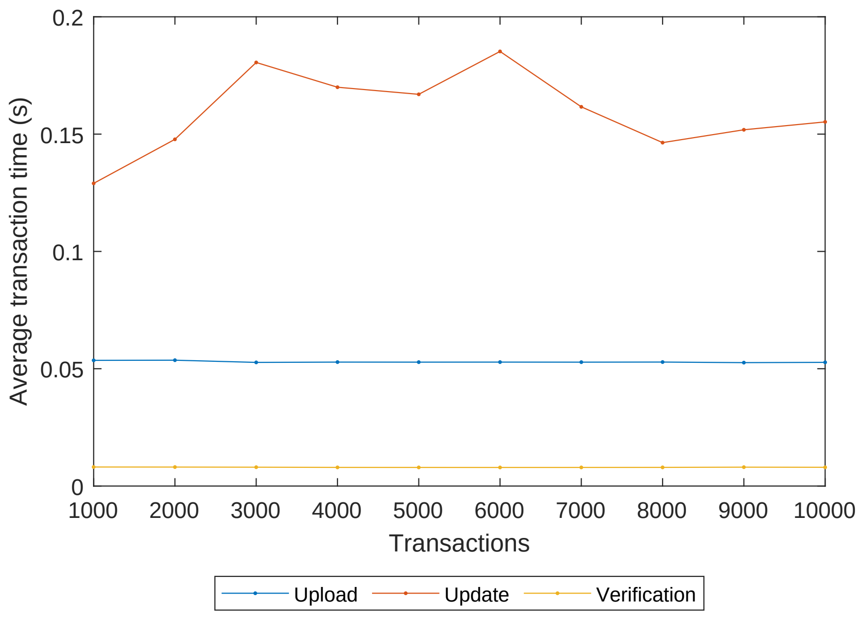

Figure 8. The average transaction time for upload, update, and verification.

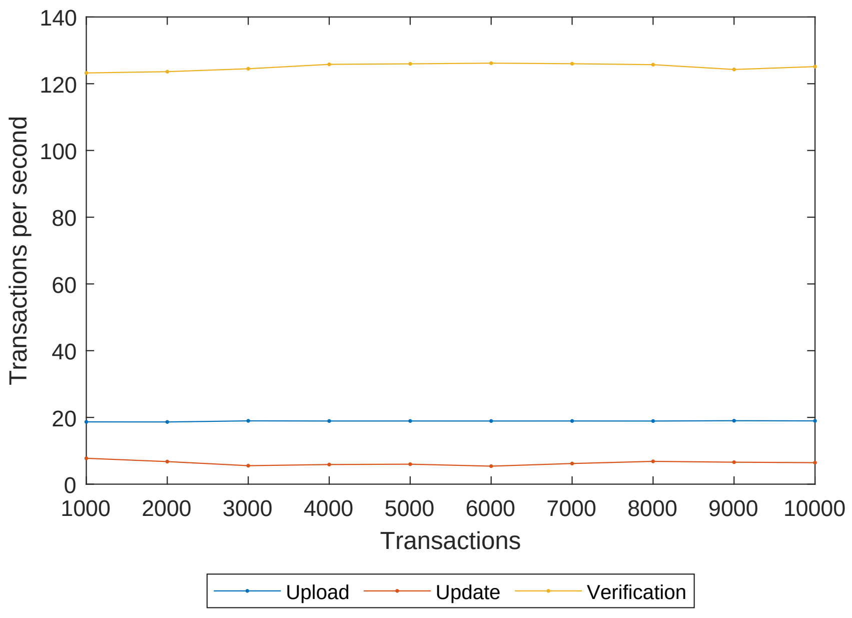

Figure 9. The TPS for upload, update, and verification.

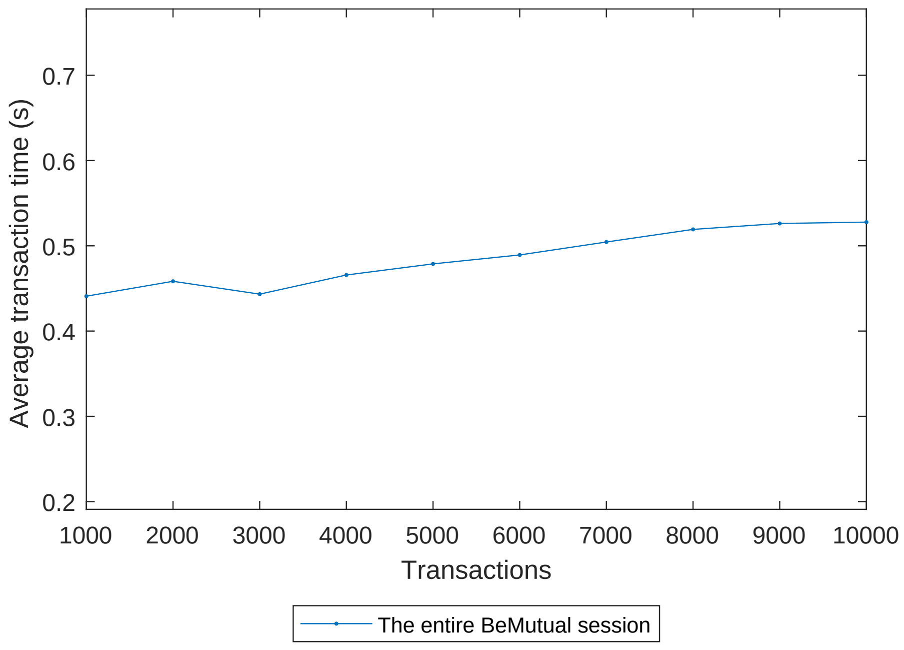

Figure 10. The average transaction time for the entire BeMutual session.

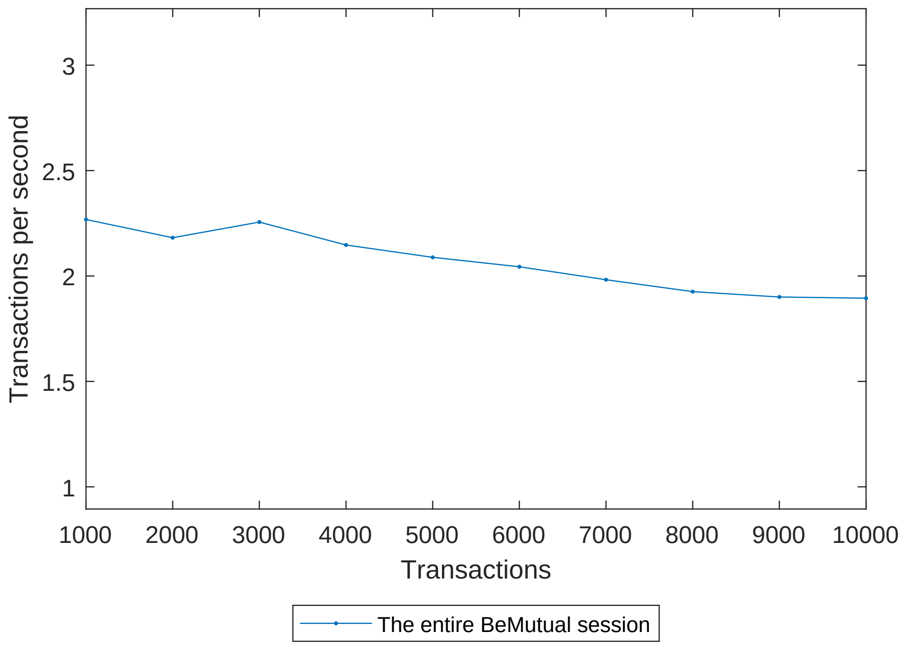

Figure 11. The TPS for the entire BeMutual session.

The results yield the following insights:

1. For the average transaction time, the order of magnitude is $verification < upload < update$, while the corresponding order for TPS is reversed.
   Both upload and update involve writing data to the BeDNS blockchain. However, the update operation also involves querying, and hence the average transaction time for update is greater than that for upload. Conversely, validation does not involve data writing, only data querying and retrieval, and hence the average transaction time is minimal, with the opposite magnitude relationship for TPS.
2. The line graphs of the average transaction time for upload and verification are relatively flat while the line graph of the average transaction time for update fluctuates more.
   The process for the upload operation is relatively fixed, resulting in a smoother line.

Although the verification operation requires querying, which may vary in time, the average transaction time is inherently short because no data writing is required, making fluctuations less noticeable in the graph. The upload operation's average transaction time is relatively longer, and, due to the variability in querying time, the line appears more fluctuating.

3. The average transaction time of the entire BeMutual session shows a slight increase as the number of iterations increases.
   The entire BeMutual session encompasses multiple basic transactions, resulting in a longer total time compared to a single transaction (upload, update, verification). As the total number of iterations for the entire BeMutual session increases, hardware platform limitations cause a backlog, leading to an increase in the average transaction time.

This simulation measured the average transaction time and corresponding TPS of three single transactions (upload, update, verification), as well as the entire BeMutual session. A secure BeMutual connection is a necessary prerequisite for inter-vehicle communication in IoV systems, making the results of this simulation highly indicative of the ATS of RSUs.

5.3. Dynamic Updating of the RSU Communication Group

As the vehicle travels on the road, changes in its physical location lead to dynamic updates in its trusted communication RSU group. Due to the mathematical equivalence between the service range of RSUs and the communication range of vehicles, the simulation, with vehicle V as the reference frame, investigates the entry and exit of RSUs within the communication range of vehicle V as it travels. This simulation measured the time required for the three most common scenarios in this process: (1) an unauthorized RSU entering the communication range, (2) an authorized RSU entering the communication range, and (3) an established RSU leaving the communication range, and thus calculating the TPS accordingly.

5.3.1. Scenario Design

In this simulation, six RSUs were configured for each scenario to simulate the dynamic updating of the RSU communication group for V while it is in motion.

- **Simulation Preparation**
  1. Utilizing Geth technology to deploy a blockchain locally and starting all RSU nodes
     This step is the same as the one in simulation B. In scenes 1 and 2, six RSU nodes are configured, with five of them set as established trusted communication RSU nodes for V while the remaining one is either an illegal (for scene 1) or legal (for scene 2) RSU node about to enter the communication range. In scene 3, all six RSU nodes initially belong to the trusted communication RSU group of V, with one of them leaving the communication range of V as it moves.
  2. Creating blockchain accounts for all RSU nodes and V
     This step differs slightly from simulation B. For the illegal RSU in scenario 1, this simulation only employs one node initiated by Geth, without subsequent mapping relationship upload operations or integration into the BeDNS blockchain.
  3. Developing a Python program to simulate the dynamic update of RSU groups
     The simulation has created a Python program for each scenario separately to facilitate debugging. The flow block diagram for the RSU's entering process is shown in Figure 12, and the one for the RSU's leaving process is generally similar, and will not be displayed.

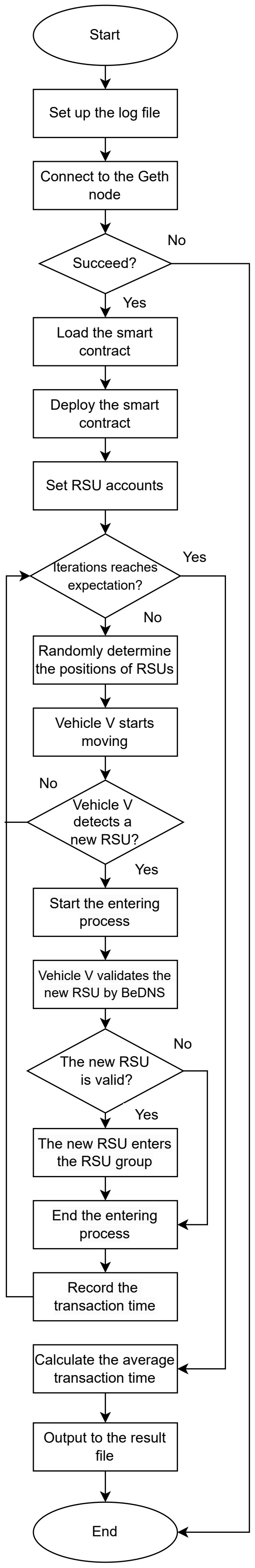
Start
Set up the log file
Connect to the Geth node
Succeed?
No
Yes
Load the smart contract
Deploy the smart contract
Set RSU accounts
Iterations reaches expectation?
Yes
No
Randomly determine the positions of RSUs
Vehicle V starts moving
No
Vehicle V detects a new RSU?
Yes
Start the entering process
Vehicle V validates the new RSU by BeDNS
No
The new RSU is valid?
Yes
The new RSU enters the RSU group
End the entering process
Record the transaction time
Calculate the average transaction time
Output to the result file
End

Figure 12. The flow block diagram for the RSU's entering process.

- **Simulation Measures**
  1. An unauthorized/authorized RSU entering the communication range
     The simulation randomly distributes the physical positions of RSUs according to a Poisson distribution to simulate V entering the communication range of an RSU node. The coordinates of the five initially established RSU nodes for trusted communication with V must meet the following conditions:
     - V should be within the service range of all RSUs at the outset.
     - V cannot leave any service range of these RSUs during the entire process.

     The coordinates of the sixth RSU node used for the simulation also need to meet the following requirements:
     - V should be out of the service range of the RSU at the outset.
     - The RSU must be located along the direction of travel of V.
     - When V is closest to the RSU, the distance between them should not exceed the radius of the RSU's service range r.

     Failure to meet any of the above conditions will result in simulation failure. To improve the success rate of the simulation while ensuring its randomness, the following procedures were implemented:
     (a) The maximum distance from the initial position of V of the first five RSUs is restricted to r.
     (b) The random distance between the initial position of the sixth RSU and the vehicle is generated within the range of $r + 1$ to $1.5r$.
     (c) The randomly generated azimuth angle concerning V's travel direction is constrained within -60 degrees to 60 degrees.
     (d) Thousands of simulations were repeated and line charts were plotted based on the results, which are described in the *Results* part in detail.

     Considering the necessity of ensuring the randomness of the simulations, the above procedures cannot eliminate the possibility of simulation failures.
     The aforementioned parameter adjustments (b) and (c) were adopted to improve the success rate of the simulation. To demonstrate that these two parameter adjustments do not affect the simulation results, a comparative simulation was conducted, which tested the average transaction time under conditions with and without the above parameter adjustments. The results of the simulation are as follows:
     - For invalid RSUs, in the case with adjustments, the average of the simulation results is 0.04210, while, without adjustments, it is 0.04708.
     - For valid RSUs with adjustments, the average of the simulation results is 0.06227, while, without adjustments, it is 0.06167.

     Based on the results above, adjustments (b) and (c) do not make obvious differences to the simulation results, which can be adopted in the simulation.
     The simulation only records the time of successful trials and displays the number of successful trials in the *Results* section.
  2. An established RSU leaving the communication range
     The random scattering rules of RSUs in this scenario are the same as in scenarios 1 and 2, and the initial coordinates of the six RSUs have the same requirements as the first five RSUs in scenarios 1 and 2.
     As V moves, one RSU will inevitably leave the communication range of V first, ensuring that simulation failures due to RSU coordinate issues do not occur as long as all RSUs are within V's communication range initially.
     When an RSU leaves the communication range of V, the number of remaining RSU nodes participating in the PBFT mechanism decreases to an odd number, 5. The odd number 5 is chosen for the following reasons:

– In the PBFT mechanism, $3f + 1$ RSUs are deployed, where f represents the maximum number of Byzantine faults that can be tolerated. The number of nodes in a PBFT network reaches its minimum value of 4, the lower bound for the number of RSUs in this simulation, when f is 1 [49].
– An even number of nodes may negatively impact the reliability performance of PBFT consensus based on the result of the simulation in [52].
– This simulation is aimed at obtaining the lower bound for the TPS of the three scenarios, and the number of nodes should be as small as possible.

Based on the reasons above, the odd number 5 is the minimal number that meets the three requirements.

5.3.2. Simulation Setups and Parameters

The basic setups and parameters are listed at the beginning of this section. In this simulation, the vehicle speed v is set to 1, the RSU service range radius r is 7, and the number of RSUs and vehicles in each scenario is 6 and 1, respectively.

5.3.3. Results

In this simulation, the performance of the three scenarios was assessed across iterations ranging from 1000 to 10,000, with a step size of 1000. The average transaction time and the corresponding TPS were computed for each scenario. The results were then graphically depicted, with the average transaction times for all three scenarios depicted in one graph as shown in Figure 13, and the TPS for all three scenarios in another graph as shown in Figure 14.

The results yield the following insights:

1. For the average transaction time, the order of magnitude is $invalid\quad RSU\quad entry <$ $valid\quad RSU\quad entry < RSU\quad exit$ while the corresponding order for TPS is reversed. In the scenario of valid RSU entry, after successful BeDNS verification, updates to the RSU group are conducted by both the relevant RSUs and vehicle V, while no such updates occur in the case of invalid RSU entry. Consequently, the average transaction time is shorter in the latter compared to the former, with a reversed relationship observed for TPS. Neither above scenarios involve view updates. In contrast, in the RSU exit scenario, where updates to the RSU group are executed and view change may also be initiated, the average transaction time is the longest, leading to minimal TPS.
2. In all three scenarios, there is a noticeable increase in the average transaction time as the number of transactions grows.
   Due to hardware platform limitations, an accumulation of transactions occurs as their quantity increases, leading to a rise in the average transaction time across each scenario.

The simulation tested the dynamic updating of the RSU communication group and measured the average transaction time and corresponding TPS for three of the most common scenarios in this process. This provides a partial basis for the expected setting of ATS in subsequent simulation D.

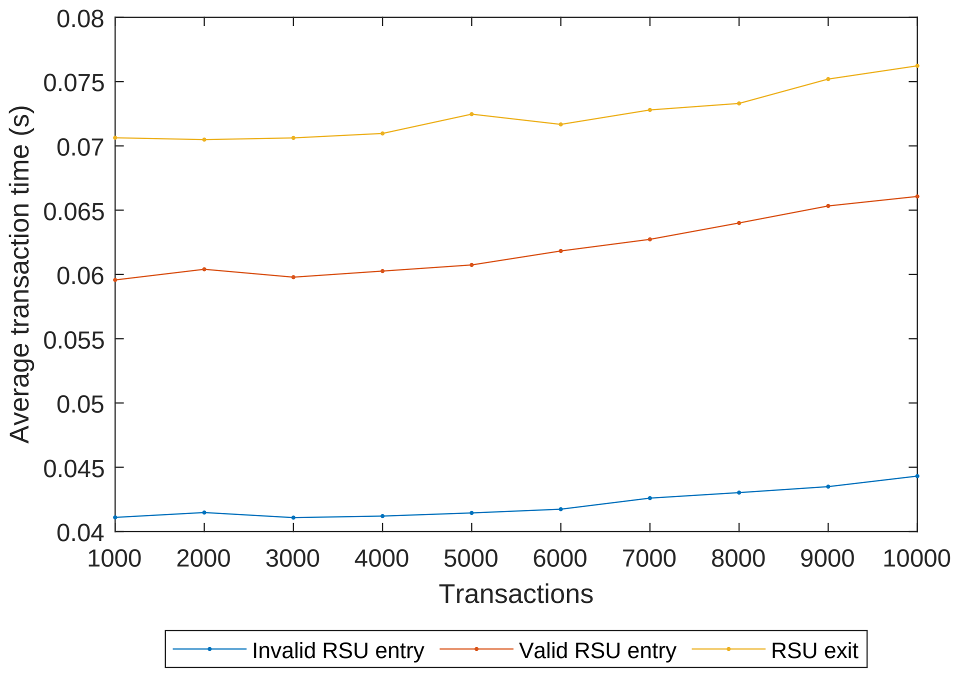

Figure 13. The average transaction time for dynamic updating of the RSU communication group.

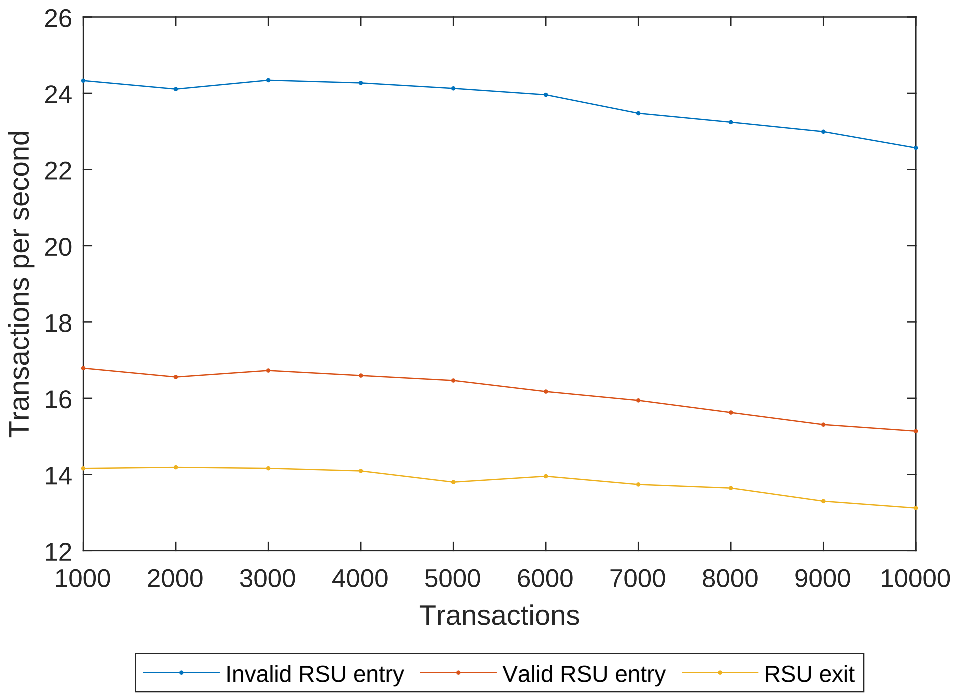

Figure 14. The TPS for dynamic updating of the RSU communication group.

5.4. Effective RSUs

The adequate length of the ATS serves as a guarantee for communication between vehicles and RSUs in this model, ensuring the completion of at least one full communication service cycle. This simulation aims to evaluate the average number of RSUs with ATS t longer than required (effective RSUs) along the way for different vehicles (van, private car, and taxi) based on simulations A, B, and C.

5.4.1. Scenario Design

- **Requirements Analysis**
  - Distribution of RSUs along the road
    The simulation first needs to simulate the distribution of RSUs on the road. Considering the complexity of real-world road conditions, simulating the distribution of RSUs along the road becomes even more challenging. To simplify the simulation of RSU distribution, this simulation first randomly generates a path and then randomly places RSUs along the path, as the RSUs far away from the road do not make a difference to the result.
    In the simulation, RSUs are randomly placed along both sides of the road at each distance interval d in the perpendicular direction, following a Poisson distribution. The parameter d needs an appropriate value based on the following reasons:
    * The distance d cannot be too small as it would result in an overly dense RSU distribution.
    * If d is too large and exceeds the RSU service radius r, vehicles passing two generated RSU sets would effectively experience two independent simulations.

    Therefore, this simulation mathematically calculates the maximum RSU spacing distance D, where vehicles passing two generated RSU sets would effectively experience two independent simulations, and sets half of this value as the d parameter for the simulation, as shown in Figure 15.
    The formula for the value of the parameter d is as follows:

$$d = \frac{\sqrt{r^2 - 1}}{2} \tag{3}$$

  - Probability turning and road generation
    In real-world scenarios, vehicles do not always travel in straight lines, which impacts the ATS of RSUs because vehicles spend more time within the service range of RSUs at turns compared to straight paths. To simulate the above scenario, turning detection is conducted at each time step t_{turn} in the simulation. Based on the vehicle type (van, private car, and taxi), probability turning judgments are made, thereby generating random roads.
  - **Simulation Measures** The overall simulation is divided into two parts: (1) random road generation and random RSU placement, and (2) simulating vehicle movement, with each part comprising half of the total simulation time T.
    * Random road generation and random RSU placement
      In the first part of the simulation, a random road is generated according to the turning rules outlined in the *Analysis* section, and RSUs are randomly placed along both sides of the road concurrently with road generation. For different vehicle types, variations in turning probabilities lead to differences in the complexity of the randomly generated roads.
    * Simulating vehicle movement
      Once the road is determined, the simulation proceeds to simulate vehicle movement on the road. The vehicle V travels at a speed of v, continuously updating its communication with RSUs within its communication range as V moves while recording the duration of continuous communication with the RSUs. At a certain moment, if the communication duration between V and a particular RSU reaches t, the number of effective RSUs at that state point is incremented by 1.

    Upon completion of the above process, the average number of effective RSUs for all state points is computed, which serves as the simulation result for this trial.

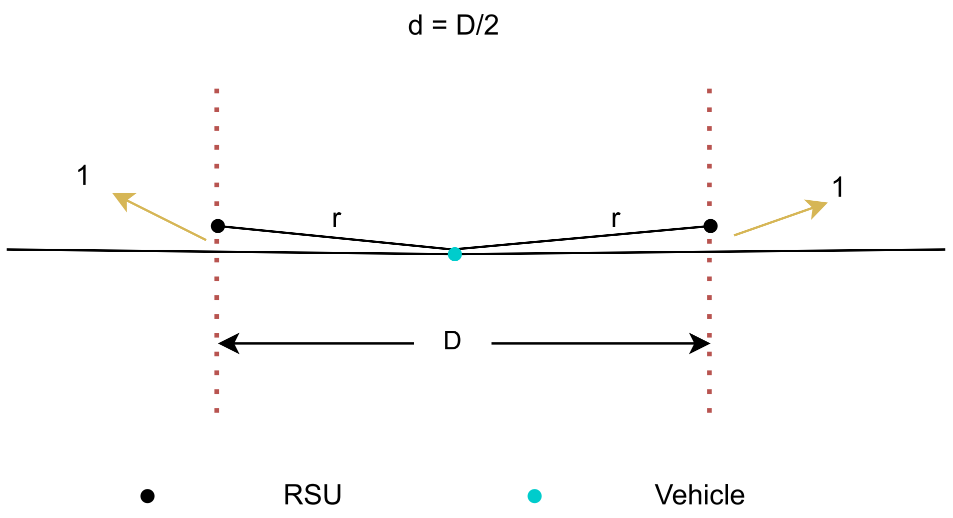

Figure 15. The mathematical model to obtain the value of D.

5.4.2. Parameters

All the parameters that may impact the result are defined and configured as shown in Table 8. The parameter d is calculated from parameter r and rounded to the nearest integer.

Table 8. Parameters for simulation D.

Parameters	**Value**	**Meaning**
v	1	The speed of V
T	1800	The total simulation time
t	5	ATS
r	7	The radius of RSUs' service range
d	7	The distance interval for RSU placement
t_{turn}	20	The time step for turning detections
λ	1	The RSU density
p_{van}	40%	The turning probability of vans
p_{car}	60%	The turning probability of private cars
p_{taxi}	80%	The turning probability of taxis

5.4.3. Results

The simulations were conducted on fifty simulated road scenarios for three types of vehicles (vans, private cars, and taxis). The resulting average effective RSU quantities are presented in Table 9, and the representative road simulation diagrams for each vehicle type are illustrated in Figures 16–18.

Table 9. The average number of effective RSUs for different vehicles.

Vehicle Type	**Average Effective RSUs**
Van	1.93
Private car	2.26
Taxi	2.61

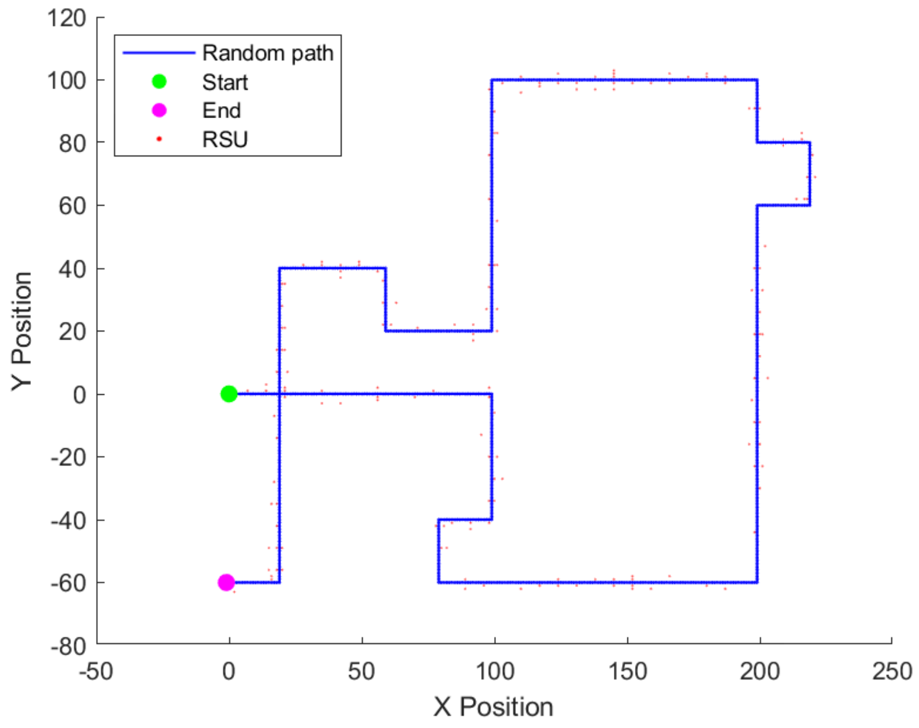

Figure 16. The representative randomly generated road of vans.

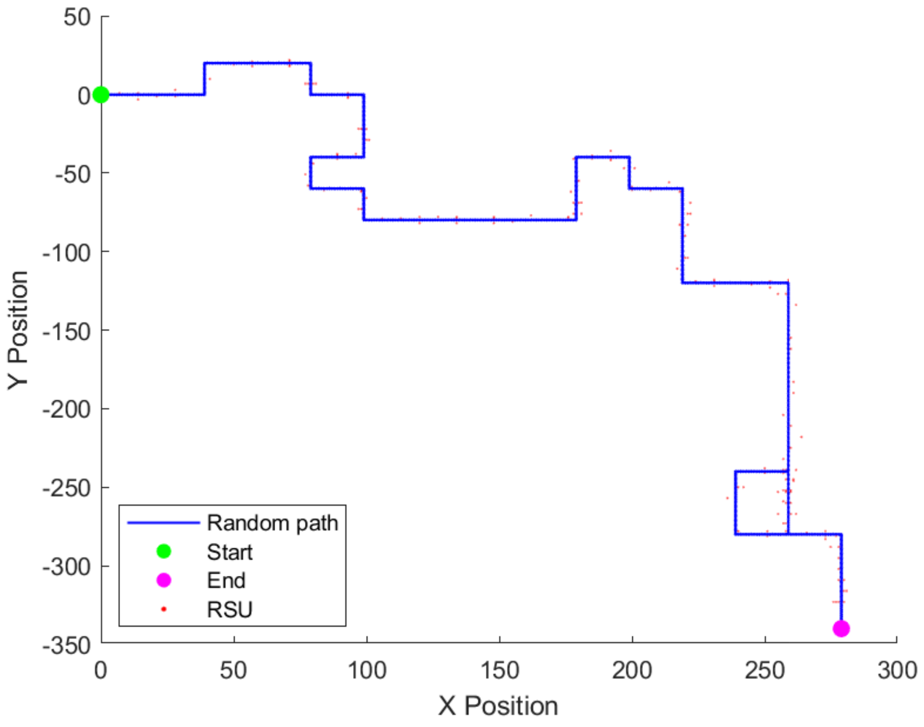

Figure 17. The representative randomly generated road of private cars.

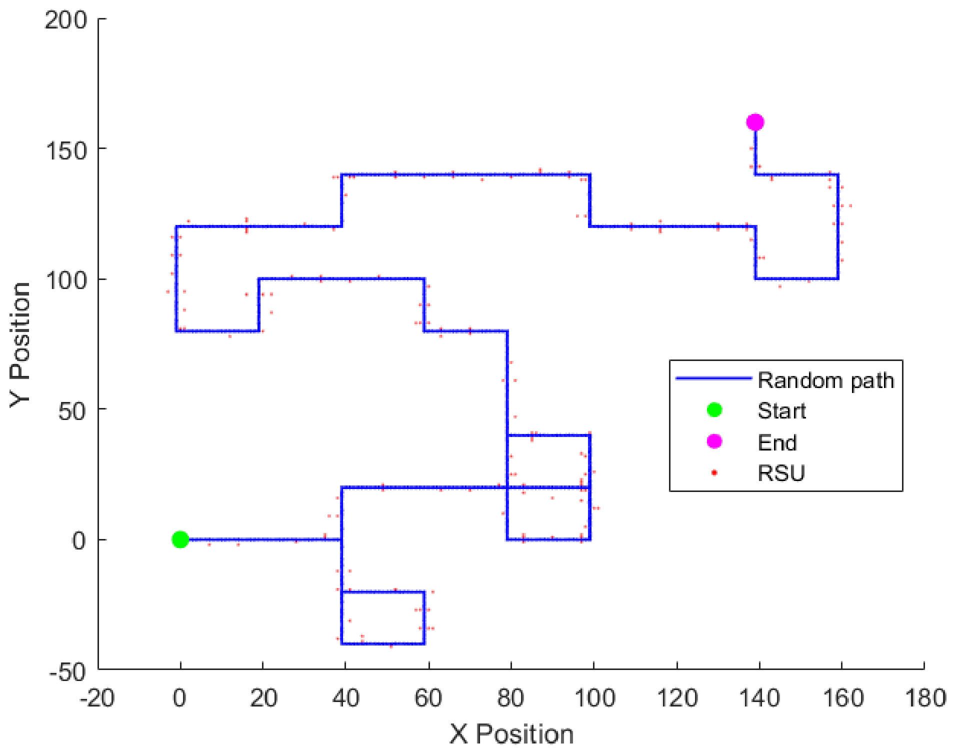

Figure 18. The representative randomly generated road of taxis.

The results yield the following insights:

1. For different types of vehicles, the higher the probability of turning, the more complex the randomly generated road conditions become.
   From vans to private cars to taxis, as the probability of turning increases, the randomly generated roads become increasingly winding. In the randomly generated roads for taxis, loops even appear.
2. The average effective RSU count follows the order $vans < private \quad cars < taxis$.
   With the probability of turning increasing, as discussed in the $Requirements \quad Analysis$ section and as mentioned in (1), the average effective RSU count obtained from simulations increases accordingly, reflecting the prolonged ATS of RSUs at vehicle turning points.

Three types of vehicles were simulated traversing roads in this simulation, and the average number of effective RSUs was recorded. This will provide data support for the deployment of RSUs in the actual implementation of this model.

6. Legal Analysis of IoV Systems: A Contract-Based Perspective

From a legal perspective, IoV systems not only enhance the performance of autonomous driving on a technical level but also make significant progress in achieving the legal objectives enshrined in digital regulations. Compared to the existing issues of privacy invasion and illegal data collection and processing faced by autonomous driving, a contract-based decentralized IoV system is more compliant with data protection laws and other digital regulations, encompassing both virtual and real entities [53]. However, at the current stage, legal and IoV systems are not yet perfect. To ensure that IoV development achieves legal and societal values, it is necessary for traditional contracts and smart contracts to work together, forming a comprehensive contractual framework.

6.1. Contractual Entities and Legal Relationships in IoV

6.1.1. Four Types of Entities and Five Types of Relationships in the Contractual System

- **Four types of entities:** As illustrated in Figure 19, these entities include RSUs, vehicles, users, and insurance companies. RSUs, or Roadside Units, function as roadside infrastructure that provides wireless services. When vehicles lack nearby RSUs or when RSUs are unavailable, they interact with edge servers as a backup through BeDNS interactions. Given the functional similarity between edge servers and RSUs and the fact that RSUs are the primary roadside infrastructure used except in special cases, the term RSUs is used to represent this category of infrastructure.

- **Five types of contractual relationships:** These include vehicle-to-RSU interactions, vehicle-to-vehicle interactions, interactions between users and vehicles, the transfer of vehicles between users, and agreements between insurance companies and users. In these five types of relationships, the first type (vehicle-to-RSUs interaction) and the second type (vehicle-to-vehicle interaction) can be facilitated through smart contracts. These contracts, combined with blockchain technology, create a data-sharing framework based on the evolution of vehicle GPS location errors. Smart contracts can enhance the accuracy and security of vehicle location correction and data sharing in the IoV. However, due to the virtual nature of smart contracts and the difficulty in modifying blockchain-based agreements, the remaining contractual relationships (user-to-vehicle, user-to-user, and insurance-to-user agreements) still rely on traditional contracts to stipulate rights and obligations. Consequently, traditional contracts and smart contracts together form the foundation of this contract-based decentralized IoV system.

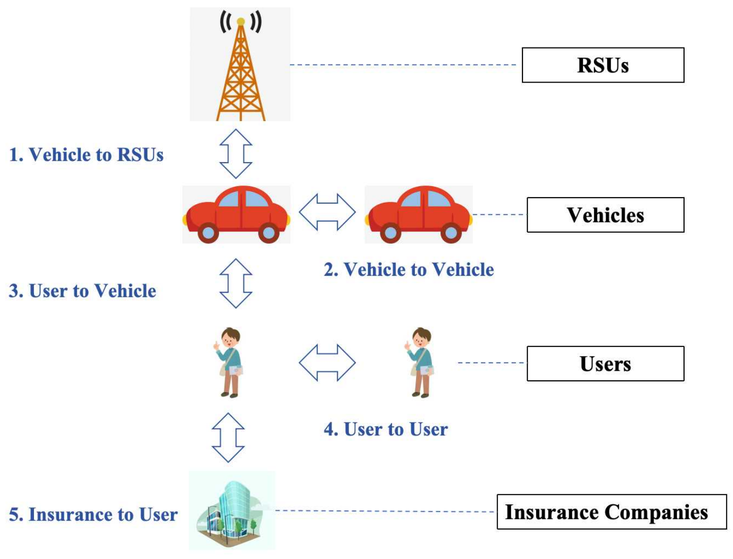

Figure 19. Four types of entities and five types of relationships in the contractual system group.

6.1.2. Legal Validity of Smart Contracts in IoV

Smart contracts possess the same legal validity as traditional contracts [54]. Despite their unique format and recognition of commitments, this uniqueness does not impede the fulfillment of obligations [55].

- **Contract Elements:** The smart contract is compatible with the theory of intent expression. Deploying a smart contract on the blockchain by both parties signifies the externalization of the contract's execution purpose, thereby producing legal effects. This aligns with the theory of intent expression in contract law. Even if the terms of the smart contract are unilaterally designed and deployed, they require the other party's consent to be valid. If not agreed upon, the other party has the right to reject the smart contract's application. Furthermore, both offers and acceptances are expressions of intent. Although these may not fully align with traditional contract norms in the context of smart contracts, they largely correspond. Smart contracts, written in computer language and executed on the blockchain, necessitate precise and concise expression due to computational costs and the nature of code. Therefore, the content conveyed by the code is undoubtedly specific and definite. In summary, smart contracts contain the essential elements of traditional contracts [56]. If the contracting parties are competent, the expression of intent is genuine, and the contract does not violate laws, smart contracts should hold the same legal validity as traditional contracts.

- **Legal Validity of Smart Contracts Analogized to Standard Terms:** A standard form contract is a preliminary form of a smart contract. Essentially, a smart contract encodes certain repeatedly applicable contractual terms into code, thereby enhancing the efficiency of contract performance. Just as standard form contracts can be used repeatedly, smart contracts can also be utilized by unspecified parties on a blockchain, where they automatically monitor and enforce the contract. Most smart contracts come with pre-drafted content that does not allow for negotiated changes. Consequently, the legal validity of smart contracts is similar to that of standard terms. For clauses in smart contracts that significantly affect both parties' interests, there should be an obligation to provide notice and explanation.

6.2. Legal Advantages of IoV in Autonomous Driving

6.2.1. IoV in Privacy Protection

The Internet of Vehicles (IoV) utilizes blockchain technology to offer wireless interactive services, thereby minimizing the collection of excessive personal data from users. This approach ensures compliance with data protection legislation. In arguing for the data law compliance of IoV, this paper primarily uses the GDPR as the data law model to demonstrate that IoV can achieve legal compliance across different jurisdictions. Given the impracticality of comprehensively discussing the data legal systems of every country and region, choosing the GDPR is the most representative and effective way to overcome legal differences across various jurisdictions. The GDPR, as a relatively advanced and comprehensive data law, serves as a benchmark for global data protection legislation, with many other data protection laws following similar principles and frameworks. For instance, China's Personal Information Protection Law (PIPL) [57] and Brazil's General Data Protection Law (LGPD) [58] share similarities with the GDPR. Therefore, under the legal framework of the GDPR, IoV can achieve data compliance and address the previous legal shortcomings of the technology, ensuring that IoV is compliant in the majority of legal jurisdictions and capable of overcoming the legal differences posed by varying legal systems.

- **Explicit lawful basis for data collection and processing:** In accordance with the purpose limitation principle of the GDPR, the IoV system explicitly defines the purpose of data use, data fields, types, and the accessible subject from the outset of the data collection process. This clear definition of data collection purposes is part of the user data privacy protection architecture, ensuring proper process documentation and management [59]. Additionally, according to Article 6(1) of the GDPR, the collection and processing of personal data by IoV can rely not only on individuals' consent but also on contractual necessity and legitimate interest to lawfully collect and process personal data. Similarly, according to Article 6 of China's PIPL, personal information processing must have a clear, reasonable purpose, be directly related to the processing purpose, and be conducted in a manner that minimizes the impact on individual rights. The collection of personal information should be limited to the minimum scope necessary to achieve the processing purpose and should not be excessive. The IoV system processes personal data based on clear and explicit contractual necessity and the legitimate interest of both parties. By distinguishing between internal and external communications and access to personal data by design, it restricts personal information to internal communications, avoiding the transmission of personal information to external parties, thereby complying with data protection laws' requirements.
- **Data minimization principle:** According to the data minimization principle of Article 5 of the GDPR, the IoV architecture is designed to collect and process only the necessary personal data. Firstly, based on the operating principles of IoV, it only requires

the interaction of the RSUs or edge server with the vehicle's location information, without the need to obtain additional vehicle or user information, making the data collected reasonable. Secondly, the whole decentralized architecture differentiates between internal and external communications. For external communications involving the interaction of data between vehicles and RSUs, other servers, or other vehicles, data collection is strictly limited to the necessary minimum information.

6.2.2. IoV in Preventing Unauthorized Data Access by Third Parties

- **Encrypted communications:** The IoV system employs encrypted communications both between vehicles and between vehicles and RSUs to prevent third-party access to data. This encryption spans the entire transmission process from the point of data generation to reception, ensuring that even if the data storage medium is illegally accessed, the data remain indecipherable. Moreover, access to data is tightly controlled [60]. The IoV's technical design requires users to provide private keys or other verification methods to access the system. Untrusted entities are excluded from the blockchain network, preventing unauthorized third parties from accessing the data.
- **Data security and sharing:** Under Article 32(1) of the GDPR, data controllers and processors shall implement appropriate technical and organizational measures to ensure a level of security that mitigates potential risks. As stated in Section 3, the sub-layer architecture can limit unnecessary access to personal data by design, preventing unauthorized third parties from accessing and obtaining data. Moreover, such a sub-layer design can facilitate secure data sharing because each entity can only access the data necessary for their processing needs. This selective access minimizes exposure and potential breaches in data sharing, thereby maintaining the integrity and security of the IoV system.

6.3. Deficiencies of Smart Contracts in Resolving the Balance of Rights between Equal Parties

Despite the significant improvements in privacy protection and prevention of third-party data access in the contract-based IoV system compared to previous autonomous driving systems, other legal issues remain. One persistent challenge is the balancing of interests between equal rights. While smart contracts, as a crucial component of the IoV, offer greater efficiency than traditional contracts, they cannot address issues of interest balancing when rights are equal. For example, they are not capable of determining when overtaking is permissible if road rights are equal or deciding whether it is acceptable to sacrifice the lives of pedestrians to protect the lives of passengers when life rights are equal. In complex traffic situations, smart contracts cannot promptly adapt to unforeseen road conditions. It is necessary to incorporate legal content and traditional contracts to make timely adjustments and address situations that smart contracts cannot cover.

6.3.1. Smart Contracts Lack Comprehensive Content to Adapt to All Driving Scenarios

- **They cannot contain ambiguous content:** The nature of smart contracts necessitates that rights and obligations are pre-defined and encoded, making subsequent modifications particularly challenging. This characteristic underscores the importance of ensuring the accuracy of pre-defined content [61]. Consequently, the drafting process for smart contracts must be executed with utmost caution. The automated execution of the promises contained in a smart contract, specifically their technical characteristics [62], leads to an increased significance of the contract drafting phase compared to the execution phase. To ensure stability, the content should be precise and consist predominantly of broad, general provisions applicable to a wide range of scenarios rather than ambiguous or situation-specific terms. Smart contracts are inherently limited in the scope and detail of the provisions that they can encompass, which often results in an inability to address complex scenarios requiring nuanced judgment and flexibility. However, in real-world driving environments, conflicts between equal parties' rights are often diverse and intricately complex, making it difficult to pre-determine rights and obligations for every possible situation.

- **Limitations of smart contracts in representing principle-based provisions:** Smart contracts operate through automated code execution, with the actual execution process occurring rapidly. This renders principle-based contractual provisions as difficult to be effectively represented in smart contracts. Such principle-based clauses can be negotiable, interpreted, and contextualized to be applicable, which smart contracts are unlikely to execute. In contrast, the virtual entities in smart contracts lack the capacity to fully comprehend and adapt to these principles. They execute the contract strictly according to the code and computations. As a result, smart contracts cannot adequately address the diverse legal needs that arise in autonomous driving scenarios. This includes addressing how to balance different rights of the same party in case of conflicts. Such a limitation necessitates the continued reliance on traditional contracts to supplement and resolve complex legal issues that smart contracts alone cannot manage effectively.

6.3.2. The Interpretability of Smart Contracts in Balancing Interests of Parties with Equal Rights

Unlike traditional contracts, which can be interpreted by courts [63] and adapted to unforeseen circumstances, smart contracts lack this interpretability [64]. Specifically, smart contracts translate terms of rights and obligations into data code [65], lacking the interpretability of natural language. To ensure the accuracy of smart contracts, they cannot include ambiguous statements. However, when conflicts of rights arise, virtual entities cannot flexibly address issues through the interpretation of specific content in the smart contract.

For example, an autonomous fleet management system governed by smart contracts can precisely define vehicle routes, speed limits, and priority rules. These terms are encoded to ensure automatic execution. The smart contract controls vehicle routes and speeds based on pre-set rules. When two autonomous vehicles (A and B) arrive at an intersection, the smart contract uses traffic conditions and signals to determine which vehicle should proceed first based on the predefined priority rules.

However, if both vehicles have equal priority and the traffic signal at the intersection suddenly fails, causing traffic confusion, the smart contract cannot adapt to this unforeseen situation. The accuracy of the smart contract terms means it cannot adjust in such emergencies, as it strictly follows the encoded priority rules and lacks the flexibility for judgment and adjustment.

6.3.3. Smart Contracts' Rigidity and the Difficulty in Balancing Rights

Smart contracts, leveraging blockchain technology, are immutable, but this characteristic also leads to rigidity [66]. The risk of automatically executing contract terms lies in the fact that, once enforced, it becomes difficult to make changes. This means that once a smart contract is established, the parties cannot alter its terms, limiting the scope for post-agreement negotiations. In situations where parties have equal rights, normal negotiations cannot occur automatically. This not only risks depriving the parties of their legal rights to rescind, modify, or terminate the contract but also may contradict the principles of autonomy and freedom advocated by private law.

By design, a blockchain-based immutable smart contract cannot be adjusted in the same way as a traditional contract. The immutability of blockchain contributes to the rigidity of smart contracts. Usually, once the encoded promises are set in motion, they will be executed without any possibility of exerting influence [67]. This means that once a smart contract is established, the parties find it difficult to alter its terms, limiting the scope for "post-agreement" negotiations. In situations where parties have equal rights, normal negotiations are unlikely to occur automatically.

Even if there are possibilities to modify smart contracts, changing the code and altering the smart contract as a whole is not convenient. A rather impractical solution might be for the parties to agree to reverse the smart contract afterward. In this case, the most effective way for the contracting parties is to stop and destroy the existing contract. However, this would cause system disruptions, and restarting a new smart contract would incur certain costs. The modification of a smart contract cannot be as easily achieved as with traditional contracts, and its negotiable and revisable content is limited.

6.4. Combining Smart Contracts and Traditional Contracts in IoV to Address Conflicts of Equal Rights

6.4.1. Addressing the Limitations of Smart Contracts through Traditional Contracts

The limitations of smart contracts, particularly in terms of content adaptability, should be addressed by supplementing them with traditional contracts [68]. As mentioned earlier, smart contracts are primarily applicable in vehicle-to-RSUs and vehicle-to-vehicle interactions. To bridge the gaps in smart contract content, traditional contracts can be employed to handle more complex and unforeseen scenarios by involving physical entities to fulfill these agreements.

Specifically, when smart contracts clearly define rights and obligations, they should be executed automatically according to their code. However, when situations arise that exceed the scope of the smart contract, traditional contracts should be used to flexibly manage these unexpected circumstances.

For example, consider a scenario where two vehicles (A and B) are traveling on the same road with equal rights. A smart contract can be pre-set to manage overtaking maneuvers. When vehicle A requests to overtake vehicle B, the smart contract checks for safety distance and speed difference conditions. If these criteria are met, vehicle B automatically yields, allowing vehicle A to overtake. This approach is effective for standard conditions covered by the smart contract's code. However, in an exceptional situation, such as a sudden traffic accident or road construction that requires urgent action, the smart contract may not be able to adapt quickly enough to handle the new context. Additionally, it is difficult to anticipate such situations and make comprehensive provisions to be included in the smart contract. In such cases, traditional contracts can provide guidelines for drivers on how to communicate and make decisions during emergencies to ensure safety. By integrating smart contracts for routine operations and traditional contracts for complex and variable situations, a comprehensive legal framework and operational guidance can be achieved. This contract-based legal framework significantly enhances the autonomy of the contracting parties. Even parties from different jurisdictions can exercise their autonomy, jointly negotiating ways to handle conflicts of rights. This ensures that IoV achieves legal compliance across various jurisdictions [69]. This approach ensures that smart contracts handle day-to-day operations efficiently while traditional contracts offer the necessary flexibility to address extraordinary and intricate circumstances, thereby providing full legal protection and operational direction.

6.4.2. Enhancing the Interpretability of Contracts in IoV

- **Incorporating traditional contract clauses:** Traditional contract clauses can supplement the parts of smart contracts that require interpretation, enhancing their flexibility. For instance, traditional contracts can include emergency clauses that allow for flexible adjustments during emergencies or traffic signal failures, specifying appropriate measures for handling such situations. By combining the automated execution capabilities of smart contracts with the interpretative clauses of traditional contracts, it is possible to achieve automatic processing when needed, while also providing space for human interpretation. For, example, a traditional contract could specify that in the event of a traffic signal failure, drivers must follow certain predefined procedures to ensure safety. This clause can be referenced by the smart contract to pause automatic execution and permit human intervention.

- **Implementing off-chain governance mechanisms:** Given that the entire IoV system includes not only the virtual entities of smart contracts but also physical entities, off-chain governance mechanisms [70] can be implemented through traditional contracts between these physical entities. This provides methods for resolving disputes and interpreting smart contract terms outside the blockchain [71]. Such mechanisms can include external arbitration or mediation processes, combined with traditional contracts, to address issues that smart contracts cannot handle autonomously. For instance, if a dispute arises regarding the execution of a smart contract, an external arbitration clause in a traditional contract can be invoked. This allows an independent arbitrator to review and interpret the contract terms and make a binding decision.

6.4.3. Enhancing the Flexibility of a Contract-Based IoV System

Due to the immutable nature of smart contracts, they offer high security, a characteristic that cannot be easily compromised. However, to avoid the rigidity that this immutability can bring [72], the flexibility of the entire IoV system can be enhanced through the following methods:

- **Using upgradable smart contract frameworks:** By employing upgradable smart contract frameworks such as OpenZeppelin, it is possible to update parts of the contract logic without altering the entire contract [73]. This method enhances the flexibility of smart contracts when modifications or updates are necessary. Taking the proxy contract model, for example, in this model, a permanent proxy contract points to the logic contract. When the logic needs to be changed, a new logic contract is deployed, and the proxy contract is updated to point to the new logic contract.
- **Designing modular smart contracts:** Designing modular smart contracts involves breaking down different functionalities into multiple independent contracts [74]. This approach allows specific modules to be modified or updated without redeploying the entire system.
- **Combining traditional and smart contracts:** Integrating traditional contracts with smart contracts can introduce human oversight into the execution process of smart contracts. This ensures that complex decisions can be reviewed and interpreted by humans when necessary, combining the advantages of automated execution with human judgment.

6.5. Summary

From a legal perspective, the IoV system cannot be fully implemented by relying solely on smart contracts, nor can it depend entirely on inefficient traditional contracts. Thus, smart contracts and traditional contracts should not be viewed as opposing mechanisms but as complementary ones. Together, they form the legal foundation of a contract-based decentralized IoV system. When the smart contract explicitly stipulates terms, the precise provisions in the contract should take precedence. However, in instances where conflicts arise between the two, the flexible and interpretable traditional contracts should take precedence.

7. Conclusions

This paper has demonstrated the design and implementation details of BeACONS. The tests of three typical scenarios in BeACONS and road side unit availability based on random distribution along vehicle routes have been finished, showing an enhancement in communication reliability and system responsiveness within IoV, which can provide data support for the real-world deployment of BeACONS. From a legal perspective, this paper also highlights that smart contracts and traditional contracts form the legal foundation of a contract-based decentralized IoV system to jointly coordinate and resolve conflicts of interest between parties with equal rights in the process of autonomous driving. Future aspirations include the real-world deployment of BeACONS to enhance security, reduce latency, and lessen reliance on centralized infrastructures in the IoV.

Author Contributions: Conceptualization, H.X., Q.S., Z.L. and Y.L.; methodology, G.J., Q.S. and Z.Z.; software, G.J.; validation, G.J. and Z.Z.; formal analysis, Z.Z. and G.J.; investigation, G.J.; resources, H.X., Y.L.; data curation, G.J.; writing—original draft preparation, G.J., L.X., Z.Z. and Z.L.; writing—review and editing, G.J., L.X., H.X., Z.L. and Z.Z.; visualization, G.J., L.X. and Z.Z.; supervision, Z.L., H.X. and Y.L.; project administration, Z.L., H.X. and Y.L.; funding acquisition, Y.L., Z.L. and H.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by Shanghai Customs College under project no. 2315027A2021 and the Fundamental Research Funds for the Central Universities, Tongji University, Project No. 22120240564.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author/s.

Acknowledgments: The authors would like to thank Qi Shi, Jingyi Sun, Hanwei Fu, Peizhe Fu, Jiayuan Ma, Hao Xu, and Erwu Liu, who provided the prior work: the BeACONS Framework for the study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Yang, F.; Wang, S.; Li, J.; Liu, Z.; Sun, Q. An overview of internet of vehicles. *China Commun.* **2014**, *11*, 1–15. [CrossRef]
2. Taslimasa, H.; Dadkhah, S.; Neto, E.C.P.; Xiong, P.; Ray, S.; Ghorbani, A.A. Security issues in internet of vehicles (iov): A comprehensive survey. *Internet Things* **2023**, *22*, 100809. [CrossRef]
3. Sharma, S.; Kaushik, B. A survey on internet of vehicles: Applications, security issues solutions. *Veh. Commun.* **2019**, *20*, 100182. [CrossRef]
4. El-Rewini, Z.; Sadatsharan, K.; Selvaraj, D.F.; Plathottam, S.J.; Ranganathan, P. Cybersecurity challenges in vehicular communications. *Veh. Commun.* **2020**, *23*, 100214. [CrossRef]
5. Alghamdi, T.A.; Khalid, R.; Javaid, N. A Survey of Blockchain Based Systems: Scalability Issues and Solutions, Applications and Future Challenges. *IEEE Access* **2024**, *12*, 79626–79651. [CrossRef]
6. Yue, K.; Zhang, Y.; Chen, Y.; Li, Y.; Zhao, L.; Rong, C.; Chen, L. A Survey of Decentralizing Applications via Blockchain: The 5G and Beyond Perspective. *IEEE Commun. Surv. Tutorials* **2021**, *23*, 2191–2217. [CrossRef]
7. Han, R.; Yu, J.; Lin, H.; Chen, S.; Esteves-Veríssimo, P. On the security and performance of blockchain sharding. *Cryptol. Eprint Arch.* **2021**, *2021*, 1276. Available online: https://eprint.iacr.org/2021/1276 (accessed on 11 June 2024).
8. Xu, H.; Liu, X.; Zeng, Q.; Li, Q.; Ge, S.; Zhou, G.; Forbes, R. DecentRAN: Decentralized Radio Access Network for 5.5G and Beyond. In Proceedings of the 2023 IEEE International Conference on Communications Workshops (ICC Workshops), Rome, Italy, 28 May–1 June 2023; pp. 556–561. [CrossRef]
9. Xu, H.; Sun, Y.; Zhang, X.; Liu, E.; I, C.L. When Web 3.0 Meets Reality: A Hyperdimensional Fractal Polytope P2P Ecosystems. *arXiv* **2023**, arXiv:2308.06829.
10. Baskaran, S. A Survey of Applications using Blockchain Technology. In Proceedings of the 2022 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 25–27 January 2022; pp. 1–6. [CrossRef]
11. Shen, J. Blockchain technology and its applications in digital content copyright protection. In Proceedings of the 4th International Conference on Economic Management and Green Development, Online, 28 January 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 18–25.
12. Zhou, Z.; Guo, C.; Zhang, X.; Wang, R.; Zhang, L.; Imran, M. A Blockchain-based Data Sharing Marketplace with a Federated Learning Use Case. In Proceedings of the 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Dubai, United Arab Emirates, 1–5 May 2023; pp. 1041–1044. [CrossRef]
13. Xu, H.; Sun, Y.; Li, Z.; Sun, Y.; Zhang, L.; Zhang, X. deController: A Web3 Native Cyberspace Infrastructure Perspective. *IEEE Commun. Mag.* **2023**, *61*, 68–74. [CrossRef]
14. Esposito, C.; De Santis, A.; Tortora, G.; Chang, H.C.; Choo, K.K.R. Blockchain: A panacea for healthcare cloud-based data security and privacy? *IEEE Cloud Comput.* **2018**, *5*, 31–37. [CrossRef]
15. Hasselgren, A.; Kralevska, K.; Gligoroski, D.; Pedersen, S.A.; Faxvaag, A. Blockchain in healthcare and health sciences—A scoping review. *Int. J. Med. Inform.* **2020**, *134*, 104040. [CrossRef] [PubMed]
16. Christidis, K.; Devetsikiotis, M. Blockchains and smart contracts for the internet of things. *IEEE Access* **2016**, *4*, 2292–2303. [CrossRef]
17. Castleman, R. The Applications of Blockchain in Data Management. AIIM, 2020. Available online: https://info.aiim.org/aiim-blog/the-applications-of-blockchain-in-data-management (accessed on 11 June 2024).
18. Cui, J.; Zhang, J.; Zhong, H.; Shi, R.; Xu, Y. An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks. *Inf. Sci.* **2018**, *451–452*, 1–15. [CrossRef]
19. Sharma, R.; Chakraborty, S. BlockAPP: Using blockchain for authentication and privacy preservation in IoV. In Proceedings of the 2018 IEEE Globecom Workshops (GC Wkshps), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6.

20. Ryan, M. The future of transportation: Ethical, legal, social and economic impacts of self-driving vehicles in the year 2025. *Sci. Eng. Ethics* **2020**, *26*, 1185–1208. [CrossRef]
21. Zavvos, E.; Gerding, E.H.; Yazdanpanah, V.; Maple, C.; Stein, S. Privacy and Trust in the Internet of Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 10126–10141. [CrossRef]
22. Nahri, M.; Boulmakoul, A.; Karim, L.; Lbath, A. IoV distributed architecture for real-time traffic data analytics. *Procedia Comput. Sci.* **2018**, *130*, 480–487. [CrossRef]
23. Contreras-Castillo, J.; Zeadally, S.; Guerrero-Ibañez, J.A. Internet of vehicles: Architecture, protocols, and security. *IEEE Internet Things J.* **2017**, *5*, 3701–3709. [CrossRef]
24. Li, L.; Zhao, C.; Wang, X.; Li, Z.; Chen, L.; Lv, Y.; Zheng, N.-N.; Wang, F.-Y. Three Principles to Determine the Right-of-Way for AVs: Safe Interaction With Humans. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 7759–7774. [CrossRef]
25. Uzair, M. Who is liable when a driverless car crashes? *World Electr. Veh. J.* **2021**, *12*, 62. [CrossRef]
26. Kaufman, J.J.D. Self-Driving Cars and Torts: Determining Liability in a Twenty-First Century Car Accident. *Ateneo LJ* **2019**, *64*, 1224.
27. Pétervári, K.; Pázmándi, K. Law and Economic Analysis of the Legal Environment Related to the Autonomous Vehicles—Is There a Legal Paradigm Shift? *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *448*, 012030. [CrossRef]
28. Shi, Q.; Sun, J.; Fu, H.; Fu, P.; Ma, J.; Xu, H.; Liu, E. BeACONS: A Blockchain-enabled Authentication and Communications Network for Scalable IoV. *arXiv* **2024**, arXiv:2405.08651.
29. Xu, H.; Zhang, L.; Sun, Y.; I, C.L. Be-ran: Blockchain-enabled open ran with decentralized identity management and privacy-preserving communication. *arXiv* **2021**, arXiv:2101.10856.
30. Zhou, Z.; Guo, C.; Xu, H.; Zhang, X.; Fan, Y.; Zhang, L. Be-dns: Blockchain-enabled decentralized name services and p2p communication protocol. In Proceedings of the 2023 IEEE 9th World Forum on Internet of Things, WF-IoT 2023, Aveiro, Portugal, 12–27 October 2023.
31. Kamil, I.A.; Ogundoyin, S.O. An improved certificateless aggregate signature scheme without bilinear pairings for vehicular ad hoc networks. *J. Inf. Secur. Appl.* **2019**, *44*, 184–200. [CrossRef]
32. Zhao, Y.; Hou, Y.; Wang, L.; Kumari, S.; Khan, M.K.; Xiong, H. An efficient certificateless aggregate signature scheme for the internet of vehicles. *Trans. Emerg. Telecommun. Technol.* **2020**, *31*, e3708. [CrossRef]
33. Xie, Z.; Chen, Y.; Ali, I.; Pan, C.; Li, F.; He, W. Efficient and secure certificateless signcryption without pairing for edge computing-based internet of vehicles. *IEEE Trans. Veh. Technol.* **2023**, *72*, 5642–5653. [CrossRef]
34. Genc, Y.; Aytas, N.; Akkoc, A.; Afacan, E.; Yazgan, E. Elcpas: A new efficient lightweight certificateless conditional privacy preserving authentication scheme for iov. *Veh. Commun.* **2023**, *39*, 100549. [CrossRef]
35. Mei, Q.; Xiong, H.; Chen, J.; Yang, M.; Kumari, S.; Khan, M.K. Efficient Certificateless Aggregate Signature With Conditional Privacy Preservation in IoV. *IEEE Syst. J.* **2021**, *15*, 245–256. [CrossRef]
36. Tan, H.; Zheng, W.; Vijayakumar, P. Secure and Efficient Authenticated Key Management Scheme for UAV-Assisted Infrastructure-Less IoVs. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 6389–6400. [CrossRef]
37. Ling, X.; Wang, J.; Le, Y.; Ding, Z.; Gao, X. Blockchain radio access network beyond 5G. *IEEE Wirel. Commun.* **2020**, *27*, 160–168. [CrossRef]
38. Wang, J.; Ling, X.; Le, Y.; Huang, Y.; You, X. Blockchain-enabled wireless communications: A new paradigm towards 6g. *Natl. Sci. Rev.* **2021**, *8*, nwab069. [CrossRef] [PubMed]
39. Cao, B.; Wang, Z.; Zhang, L.; Feng, D.; Peng, M.; Zhang, L.; Han, Z. Blockchain systems, technologies, and applications: A methodology perspective. *IEEE Commun. Surv. Tutorials* **2023**, *25*, 353–385. [CrossRef]
40. Xu, H.; Klaine, P.V.; Onireti, O.; Cao, B.; Imran, M.; Zhang, L. Blockchain-enabled resource management and sharing for 6G communications. *Digit. Commun. Netw.* **2020**, *6*, 261–269. [CrossRef]
41. Klaine, P.; Xu, H.; Zhang, L.; Imran, M.; Zhu, Z. A Privacy-Preserving Blockchain Platform for a Data Marketplace. *Distrib. Ledger Technol. Res. Pract.* **2022**, *2*, 1–16. [CrossRef]
42. Cao, M.; Zhang, L.; Cao, B. Toward on-device federated learning: A direct acyclic graph-based blockchain approach. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 2028–2042. [CrossRef]
43. Liu, W.; Cao, B.; Peng, M. Web3 technologies: Challenges and opportunities. *IEEE Netw.* **2023**, *38*, 187–193. [CrossRef]
44. Jabbar, R.; Kharbeche, M.; Al-Khalifa, K.; Krichen, M.; Barkaoui, K. Blockchain for the internet of vehicles: A decentralized iot solution for vehicles communication using ethereum. *Sensors* **2020**, *20*, 3928. [CrossRef] [PubMed]
45. Vishwakarma, L.; Nahar, A.; Das, D. Lbsv: Lightweight blockchain security protocol for secure storage and communication in sdn-enabled iov. *IEEE Trans. Veh. Technol.* **2022**, *71*, 5983–5994. [CrossRef]
46. Alladi, T.; Naren; Bansal, G.; Chamola, V.; Guizani, M. SecAuthUAV: A Novel Authentication Scheme for UAV-Ground Station and UAV-UAV Communication. *IEEE Trans. Veh. Technol.* **2020**, *69*, 15068–15077. [CrossRef]
47. K, R.; Pathak, D.; Tammana, P.; A, A.F.; Alladi, T. Accelerating PUF-based UAV Authentication Protocols Using Programmable Switch. In Proceedings of the 2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 4–8 January 2022; pp. 309–313. [CrossRef]
48. Castro, M.; Liskov, B. Practical Byzantine Fault Tolerance. In Proceedings of the 3rd Symposium on Operating Systems Design and Implementation (OSDI 99), New Orleans, LA, USA, 22–25 February 1999.

49. Castro, M.; Liskov, B. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.* **2002**, *20*, 398–461. [CrossRef]
50. G.E. Developers. Go Ethereum Documentation. Available online: https://geth.ethereum.org/docs (accessed on 11 June 2024).
51. Web3.py. Web3.py Documentation. 2024. Available online: https://web3py.readthedocs.io/en/stable/ (accessed on 9 June 2024).
52. Xu, H.; Fan, Y.; Li, W.; Zhang, L. Wireless Distributed Consensus for Connected Autonomous Systems. *IEEE Internet Things J.* **2023**, *10*, 7786–7799. [CrossRef]
53. Li, Z.; Xu, H.; Fang, Y.; Zhao, B.; Zhang, L. Recordism: A social-scientific prospect of blockchain from social, legal, financial, and technological perspectives. *arXiv* **2023**, arXiv:2204.00823.
54. Liu, Y.; Huang, J. Legal creation of smart contracts and the legal effects. *J. Physics Conf. Ser.* **2019**, *1345*, 042033. [CrossRef]
55. Ferreira, A. Regulating smart contracts: Legal revolution or simply evolution? *Telecommun. Policy* **2021**, *45*, 102081. [CrossRef]
56. Catchlove, P. Smart Contracts: A New Era of Contract Use. *SSRN* **2017**, 3090226. Available online: https://ssrn.com/abstract=3090226 (accessed on 11 June 2024).
57. Tan, Z.; Zhang, C. China's PIPL and DSL: Is China following the EU's approach to data protection? *J. Data Prot. Priv.* **2021**, *5*, 7–25.
58. Gadoni Canaan, R. The effects on local innovation arising from replicating the GDPR into the Brazilian General Data Protection Law. *Internet Policy Rev.* **2023**, *12*. . [CrossRef]
59. Mulder, T.; Vellinga, N.E. Exploring data protection challenges of automated driving. *Comput. Law Secur. Rev.* **2021**, *40*, 105530. [CrossRef]
60. Zhang, Y.; Kasahara, S.; Shen, Y.; Jiang, X.; Wan, J. Smart contract-based access control for the internet of things. *IEEE Internet Things J.* **2018**, *6*, 1594–1605. [CrossRef]
61. Raskin, M. The law and legality of smart contracts. *Geo. Tech. Rev.* **2016**, *1*, 305.
62. Alharby, M.; Van Moorsel, A. Blockchain-based smart contracts: A systematic mapping study. *arXiv* **2017**, arXiv:1710.06372.
63. Vatiero, M. Smart contracts vs incomplete contracts: A transaction cost economics viewpoint. *Comput. Law Secur. Rev.* **2022**, *46*, 105710. [CrossRef]
64. Zirar, A.; Jabbar, A.; Njoya, E.; Amoozad Mahdiraji, H. Smart contract challenges and drawbacks for SME digital resilience. *J. Enterp. Inf. Manag.* 2024, *ahead of print*. [CrossRef]
65. Woebbeking, M.K. The impact of smart contracts on traditional concepts of contract law. *J. Intellect. Prop. Inf. Technol. Law* **2019**, *10*, 105.
66. Nzuva, S. Smart contracts implementation, applications, benefits, and limitations. *J. Inf. Eng. Appl.* **2019**, *9*, 63–75.
67. Werbach, K.; Cornell, N. Contracts ex machina. *Duke Lj* **2017**, *67*, 313.
68. Huckle, S.; Bhattacharya, R.; White, M.; Beloff, N. Internet of things, blockchain and shared economy applications. *Procedia Comput. Sci.* **2016**, *98*, 461–466. [CrossRef]
69. Savelyev, A. Contract law 2.0: 'Smart' contracts as the beginning of the end of classic contract law. *Inf. Commun. Technol. Law* **2017**, *26*, 116–134. [CrossRef]
70. Reijers, W.; Wuisman, I.; Mannan, M.; De Filippi, P.; Wray, C.; Rae-Looi, V.; Cubillos Vélez, A.; Orgad, L. Now the code runs itself: On-chain and off-chain governance of blockchain technologies. *Topoi* **2021**, *40*, 821–831. [CrossRef]
71. Cao, S.; Miller, T.; Foth, M.; Powell, W.; Boyen, X.; Turner-Morris, C. Integrating on-chain and off-chain governance for supply chain transparency and integrity. *arXiv* **2021**, arXiv:2111.08455.
72. Sklaroff, J.M. Smart contracts and the cost of inflexibility. *Univ. Pa. Law Rev.* **2017**, *166*, 263.
73. Amri, S.A.; Aniello, L.; Sassone, V. A review of upgradeable smart contract patterns based on openzeppelin technique. *J. Br. Blockchain Assoc.* **2023**. Available online: https://jbba.scholasticahq.com/article/73752.pdf (accessed on 10 June 2024).
74. Chen, G.; Li, H.; Liu, M.; Hsiang, S.M.; Jarvamardi, A. Knowing what is going on—A smart contract for modular construction. *Can. J. Civ. Eng.* **2023**, *50*, 210–223. [CrossRef]