Beyond Liability: Decentralized Forensics for Autonomous Vehicle Events via a Redactable Blockchain Approach

Hao Guo, *Member, IEEE,* Wanxin Li*, *Member, IEEE,* Collin Meese, *Student Member, IEEE,* Yetong Wang, and Mark Nejad, *Senior Member, IEEE*

***Abstract*—Autonomous vehicles (AVs) can sense their environment and navigate without human input. However, when AVs are involved in accidents with other AVs or human subjects, liability must be indisputably determined based on accident forensics. However, existing methods rely on centralized authorities to collect and manage accident data, making them vulnerable to cybersecurity attacks (e.g., tampering, denial-of-service). Decentralized systems, such as blockchain networks, present a promising alternative; however, their data immutability properties present challenges in practice. Consequently, this paper introduces a redactable blockchain solution for vehicular event forensics. This novel approach, underpinned by a decentralized attribute-based chameleon hash function, supports modifying data from multiple blocks, as well as individual transaction-level modification operations, without the issue of a single point of failure. We implemented the proposed system using the Charm cryptography library and the Hyperledger Fabric blockchain platform. The Chameleon hash function can support redactable operations for on-chain vehicular event records with a latency of seconds. We also conducted extensive experiments on blockchain performance and the witness group formation cost, highlighting that the proposed redactable blockchain is efficient in practice for vehicular event forensics tasks.**

***Index Terms*—Redactable Blockchain, Decentralized Attribute-based Chameleon Hash, Vehicular Forensics, Hyperledger.**

I. Introduction

Connected and Autonomous vehicles (CAVs) have experienced significant research and development attention in recent years, with companies transitioning away from controlled testing tracks to deploying their CAVs in real-world transportation systems without human operators [1]. Companies, including Waymo in the United States and Baidu in China, have recently deployed driverless, level 4 (L4) robotaxis in urban environments with the help of local government institutions [2]. However, as more AVs enter the transportation network and come into contact with human drivers, the safety implications of CAVs are becoming a critical concern for the public, local authorities, and regulators [3]. For example, as of July 3rd, 2025, there have been 833 AV collision reports documented in the state of California alone, prompting the government to enact more stringent reporting and transparency requirements [4].

When CAVs are involved in accidents between themselves or with human drivers, liability must be indubitably decided based on accident forensics. In scenarios without CAVs, this usually requires the involvement of law enforcement to act as the trusted authority, take statements, and ultimately determine the truthful facts surrounding the incident to assign liability. However, when CAVs are involved, the necessary information to determine fault will be automatically captured by the vehicle itself, nearby vehicles, and Intelligent Transportation Systems (ITS) sensors (e.g., smart camera). This presents challenges for determining liability in CAV-related incident scenarios, as parties have competing incentives to tamper with recorded events to avoid punitive or financial penalties. Consequently, it is critical for the integrity of the recorded event data to be trustworthy, the facts to be easily verifiable, and the contained information to be accessible to stakeholders securely and transparently. Traditional centralized data storage systems, such as cloud databases, are unsuitable for this application because they are owned and managed by a single authority, making them more vulnerable to cybersecurity threats and presenting trust issues due to the lack of data transparency.

Instead, blockchain technology manifests itself as a potentially transformative solution for addressing the aforementioned challenges in CAV accident forensics. Since its inception with the introduction of the Bitcoin digital cryptocurrency system by Nakamoto in 2008 [5], blockchain has become a foundational technology in the digital realm. The core characteristics of blockchains can be synthesized using the acronyms *TRUE* and *DAO*. *TRUE* encapsulates the attributes of being *Trustable*, *Reliable*, *Usable*, and *Effective*, representing the robust and secure nature of the system. On the other hand, *DAO* delineates its operational model as being *Distributed and decentralized*, *Autonomous and automated*, and *Organized and ordered*, highlighting the intricate yet organized fashion in which blockchain operates without a central authority.

Blockchain networks come in various forms and can be classified as either permissioned, public, or hybrid based on the read and write data access of users and organizations [6],

Manuscript received March 01, 2025; revised July 19, 2025; accepted October 04, 2025. This work is partially supported by the Natural Science Basic Research Program of Shaanxi (Program No.2025JC-YBMS-688), the Key R&D Programs of Taicang, 2024 under Grant TC2024SF10, XJTLU Research Development Fund under Grant No. RDF-22-02-106, and the Jiangsu Province Science and Technology Youth Talent Promotion Program under Grant No. JSTJ-2025-144. *(Corresponding author: Wanxin Li.)*

Hao Guo is with the School of Software, Northwestern Polytechnical University, Xi'an, 710129, China (e-mail: haoguo@nwpu.edu.cn).

Wanxin Li and Yetong Wang are with the Department of Communications and Networking, Xi'an Jiaotong-Liverpool University, Suzhou, 215123, China (e-mail: wanxin.li@xjtlu.edu.cn, yetong.wang19@alumni.xjtlu.edu.cn).

Collin Meese and Mark Nejad are with the Department of Civil and Environmental Engineering, University of Delaware, Newark, Delaware, 19716, USA (e-mail: {cmeese, nejad}@udel.edu).

[7]. In a permissioned or consortium blockchain, a consortium of stakeholders (e.g., individuals, companies, regulators) governs membership, consensus participation, and data access rights. In contrast, a public blockchain, such as Bitcoin, offers unrestricted network access and participation in block consensus. In this study, we propose a decentralized, secure, and modifiable autonomous vehicle event recording system based on a permissioned blockchain.

Related research in CAV networks [8] and ITS [9] have demonstrated that permissioned blockchain network technology is a promising candidate for implementing data communication and storage protocols in smart transportation. It provides numerous operational and security benefits, including decentralized and trustless data storage, programmable access controls, flexible consensus algorithms, and tamper-resistance. Existing ITS are equipped with roadside units (RSU) that serve as the communication gateway for CAVs and edge devices, as well as numerous intelligent sensors (e.g., AI-equipped cameras, LiDAR) that continuously collect data and monitor the network conditions [10].

In a blockchain-based CAV forensics system, in the event of an accident, data can be automatically collected from nearby witness vehicles, ITS sensors, and the CAV itself to construct and securely store the vehicle accident record in near-real-time without relying on a centralized intermediary to serve as the trusted authority. Due to the secure and programmable data access controls, this design helps prevent potential data manipulation and tampering while also providing authorized stakeholders (e.g., involved parties, insurance companies, government agencies) with access to the data. In other words, the decentralized blockchain ledger serves as an unbiased source of truth, ensuring impartiality and minimizing conflicts between the parties involved. In addition, the lack of reliance on a singular authority or a centralized data repository minimizes vulnerabilities and potential single points of failure, guaranteeing robustness in the system.

However, while securely and automatically recording such data is essential for accurate and immediate forensic analysis, it is also notable that not all recorded data should remain immutable indefinitely. In practice, erroneous or incomplete accident reports may be recorded in the system due to sensor or communication failures, software issues, or malicious actors. In these cases, innocent drivers may be unduly stigmatized and suffer unjust financial or punitive consequences as a result. Moreover, accidents that once marred a driver's record may no longer be relevant, and their continued existence on the decentralized blockchain can result in excessive storage and resource consumption as well as present data privacy concerns for individuals who wish to have irrelevant records purged. In such scenarios, the ability to modify and delete the erroneous or expired accident reports becomes a functional system requirement.

Consequently, the immutability of the data records stored on blockchain becomes a limitation in this case, hindering its usefulness in the application of autonomous vehicle forensics despite the other cited benefits. To address these challenges in many practical applications, the concept of redactable blockchains has been proposed, where authorized stakeholders can reach consensus on modifying or removing existing data without compromising the associated security or integrity properties.

In redactable blockchain systems, the dual objectives of data integrity and individual rights, especially the "Right to be Forgotten," can cohesively coexist. This ensures that while CAVs operate in a transparent and accountable ecosystem [11], they also respect and uphold the privacy rights of individuals (e.g., General Data Protection Regulation of the European Union), striking a delicate balance between utility and privacy. As data privacy laws, such as the General Data Protection Regulation of the European Union, continue to change and become more stringent, it is necessary to develop new approaches that can comply with the evolving digital landscape. That being said, the concept is still very new, and further research is needed to address the challenge associated with devising mechanisms to allow controlled, yet secure, modifications to the blockchain to mitigate harmful data propagation [12], [13].

Summarily, redactable blockchains improve upon traditional blockchain systems by alleviating some drawbacks from the following perspectives:

- **Scalability and storage efficiency:** As blockchains grow, storage and computational resources become a significant concern. Redactable blockchains can enhance scalability and storage efficiency by enabling the removal of outdated, irrelevant, or redundant data without compromising the blockchain's integrity and security.
- **Adaptability to changing regulations and policies:** Permissioned blockchains are often used in regulated industries, where rules and policies may change over time. Redactable blockchains offer the flexibility to adapt to these changes by allowing the modification or removal of data that no longer complies with the latest regulations or policies.
- **Error correction and dispute resolution:** In traditional blockchains, once a transaction is added to the ledger, it cannot be modified or removed, even in the case of errors or disputes. Redactable blockchains enable the correction of errors and resolution of disputes by granting designated authorities the power to modify or delete specific transactions, thereby ensuring the overall accuracy and reliability of the blockchain for vehicular event forensics.
- **Data privacy and compliance:** Data privacy regulations, such as GDPR and HIPAA, mandate the protection of personal data and the "right to be forgotten." Traditional permissioned blockchains, where data is immutable, do not allow for the removal of personal information when required by law [14]. Redactable blockchains address this knowledge gap by enabling data redaction, allowing permissioned blockchain networks to comply with data privacy regulations.

A. Our Contributions

This paper proposes a blockchain-based event recording scheme that uses the redactable mechanism to incorporate autonomous vehicles into a trusted and verifiable event recording and forensics system. In summary, it makes the following contributions:

- We propose an event recording system for vehicular networks. The system provides AVs with a novel design inspired by blockchain technology, which maintains vehicular accident events as digital records. We propose the decentralized attribute-based chameleon hash (DACH) scheme (belongs to Public Key Infrastructure (PKI) systems), where multiple attribute-based authority nodes generate the key without the trusted party. We then extend the DACH scheme to support both multiple block redaction and transaction-level redaction operations.
- We propose a new structure that links the redaction history of the blocks to enable redaction of multiple, consecutive blocks, while also supporting transaction-level data redactions within a block. To ensure security and achieve consensus during the redaction process between the stakeholders, our scheme requires T-of-N attribute authorities to reach agreement on any data redactions. We also propose a witness group formation protocol to address the threats posed by malicious connected vehicles in the accident forensics data recording process, while also ensuring only the nearby vehicles are considered as witness vehicles.
- We conduct experiments on the redactable blockchain and compare it with the Ateniese [15] scheme. The proposed system is based on the Charm cryptography library. The blockchain performance is measured by the Hyperledger Fabric and Caliper tool. Experimental results indicate that the proposed redactable blockchain is efficient in practice for vehicular event forensics tasks. Hash generation time remains constant around 0.02s for 1024-bit and 0.12s for 2048-bit, while collision generation time ranges from 0.23-0.47s and 1.34-1.62s respectively.

The rest of the paper is organized as follows. We discuss preliminary knowledge in Section II. In Section III, we describe the related work and provide a comparison table with existing approaches. In Section IV, we present the system architecture, which primarily consists of a permissioned blockchain network, witness group formation, a decentralized attribute-based chameleon hash, and redactable algorithms. In Section V, we discuss the security analysis and propose new theorems. Experiments and evaluations are presented in Section VI. Section VII concludes the paper and points out future research directions.

II. Preliminary Knowledge

A. Blockchain Structure

In this section, we provide an overview of blockchain networks, including Bitcoin and the Ethereum blockchain. A blockchain comprises many blocks, each consisting of two parts: the block header and the block body. The block header mainly consists of two parts, m_root, and p_hash, where the former is the summary of the block body, and the latter is the header's hash value of the preceding block [16]. The body includes a number of transactions. All blocks form a chain by p_hash, and the index of a block is called the height (the genesis block's height is 1).

p_hash is computed through a hash function, such as the SHA-256, to ensure that no participant can modify the blocks. l_hash stores the header's hash value of the block that is redacted to the current block; it can be computed through the $DACH$ function. Also, all transactions in the body constitute the Merkle hash tree, and the m_root ensures that no one can modify the transactions.

B. Attribute-based cryptography

Attribute-based cryptography (ABC) uses keys associated with attributes to control access to encrypted data, offering fine-grained access control compared to traditional encryption [17]. Instead of using a single key for all users, ABC allows for different keys based on user attributes and access policies, enabling more flexible and efficient access management in distributed systems.

Key concepts include the following items. Attributes: These are characteristics associated with users or data, like "country: USA", or "subscription: premium." Access Policies: These define the attributes required to decrypt data. Secret Keys: Users receive secret keys based on their attributes and the access policies they need to fulfill. Ciphertexts: Data is encrypted with attributes that define who can decrypt it.

There are two types of Attribute-Based Encryption. Key-Policy Attribute-Based Encryption (KP-ABE): The ciphertext is associated with attributes, and the secret key is associated with an access policy. Ciphertext-Policy Attribute-Based Encryption (CP-ABE): The ciphertext is associated with an access policy, and the secret key is associated with attributes [18].

C. Chameleon Hash Functions

The chameleon hash was first proposed by Krawczyk and Rabin in 1998 [19]. Chameleon hash functions mainly have four algorithms:

Key Generation Algorithm:$(Key\_Gen(1^{\lambda})) = (pk, sk)$. Given a security parameter λ, this algorithm outputs the chameleon hash public key pk and the private key sk (trapdoor). A trapdoor hash function has a secret key or "trapdoor" that, when possessed, allows for easy computation of preimages or collisions that would be practically impossible to find otherwise.

Chameleon Hash Generation Algorithm:$(ChameleonHash\_Gen(pk, m, r)) = (h)$. Input with the public pk, random number r, and the arbitrary message m, this algorithm generates the chameleon hash value h, where $h = H(m, r)$;

Chameleon Hash Verification Algorithm:$(ChameleonHash\_Ver(pk, m, h, r) = \{0,1\}$. Input with the public key pk, arbitrary message m, hash value h, and the random number r. If the h is a valid hash value, this algorithm outputs 1; otherwise outputs 0.

Chameleon Hash Collision Algorithm:$(Chameleon\_Collision(sk, m, m', h, r))$. Input with the private key sk (trapdoor), message m, new message m', the hash value h and random number r, it outputs a new random number r', which satisfies that ($ChameleonHash\_Ver(pk, m, h, r)$ = ($ChameleonHash\_Ver(pk, m', h, r')$) = 1.

Chameleon hash functions have the following security requirements:

1) Anti-collision: In the absence of trapdoor sk, we have the inputs of message m_1, m_2, and the random number r_1. It is impossible to find the new random number r_2 which satisfies $Hash(m_1, r_1) = Hash(m_2, r_2)$.

2) Trapdoor collision: When we have the trapdoor sk, for any arbitrary m_1 and r_1, given the message m_2, we can compute r_2 which satisfies $Hash(m_1, r_1) = Hash(m_2, r_2)$.

3) Semantic security: For any arbitrary message m_1 and m_2, their hash information $Hash(m_1, r_1)$, and $Hash(m_2, r_2)$ are computationally indistinguishable.

D. Redactable Blockchain

In 2017, Ateniese et al. [15] introduced the first chameleon hash functions into the blockchain using smart contracts. Redaction methods are used to modify or remove specific information from the blockchain while maintaining its integrity and security. The smart contract-based redaction method and the chameleon hash-based redaction mechanism are primary redaction techniques. Smart contract-based redaction is more flexible and customizable but may have security concerns, while chameleon hash-based redaction is more secure but less flexible. The choice between these methods depends on the specific use case, the desired level of flexibility, and the system's security requirements. We listed the differences from the following three perspectives.

From the implementation perspective: a) Smart contract-based redaction: This method uses smart contracts to enforce redaction rules. The smart contracts are designed to encode the specific conditions under which data can be redacted or modified, and they automatically execute when these conditions are met. Smart contracts are typically implemented on platforms that support Turing-complete programming languages (e.g., Ethereum). b) Chameleon hash-based redaction: This method uses a particular type of hash function called a chameleon hash. Chameleon hashes have the unique property of being collision-resistant but allowing for controlled collisions. This means that under certain conditions, two inputs can produce the same hash value. Chameleon hash-based redaction replaces the original data with the redacted data while ensuring the modified block has the same hash value.

From the flexibility perspective: a) Smart contract-based redaction: This method provides greater flexibility as the rules and conditions for redaction can be programmed directly into the smart contract. It allows for more complex and customizable redaction scenarios. b) Chameleon hash-based redaction: This method is less flexible because it relies on the properties of chameleon hash functions, which are not as versatile as smart contracts. However, it remains suitable for simple redaction scenarios and can be highly effective.

From the security perspective: a) Smart contract-based redaction: The security of smart contracts depends on the platform they are built on and the quality of the code. Vulnerabilities in the smart contract or the underlying platform could compromise the redaction process. b) Chameleon hash-based redaction: This method is considered more secure, as it relies on the cryptographic properties of chameleon hashes, which are generally resistant to attacks. However, the security of chameleon hash-based redaction also depends on the strength of the cryptographic algorithms and their implementation.

III. Related Work

We analyze current research on redactable blockchain schemes. The literature can be divided into centralized and decentralized schemes.

Regarding centralized schemes, Ateniese et al. [15] first proposed integrating chameleon hash functionality into blockchain networks. In their method, a centralized authority controlling a special private key is responsible for redacting transactions. Another early study by Deuber et al. [20] proposed a voting system, where any node can initiate redaction requests, and the remaining nodes can vote on whether to approve or deny the redaction. When nodes vote, they store the hashed redaction request in the newly generated blocks. Xu et al. [21] developed a redactable blockchain-based data management scheme with security, efficiency, and user accountability properties. Additionally, the authors propose and integrate a distributed trapdoor recovery mechanism to mitigate security risks and enforce data security. However, their method does not consider the scenario where a batch of multiple blockchain data redaction operations are needed, such as in the case where a faulty sensor provided erroneous data for multiple accident records or data fields. Instead, their system processes all redactions individually with less efficiency.

A cipher-policy and attribute-based chameleon hash system for redactable blockchain transactions was proposed by Derler et al. [22]. In their scheme, a trusted authority is responsible for distributing attribute-based keys to participating nodes. However, the centralized management of the system can potentially lead to a single point of failure and does not address the centralization issue. To address this, Zhang et al. [23] proposed the multiple attribute authorities to issue attribute private keys to the blockchain network. This allows users who meet access policies to perform modification operations. In another work, Huang et al. [24] adapted the RingCT protocol with multiple additions. In their proposed system, multiple tracing authorities can work together to jointly track the identity of users, while a variety of multi-grained (i.e., accumulator-level, commitment-level, transaction-level, and block-level) redaction operations on stored data can be enacted by a system manager.

Xu et al. [25] proposed a concept called a modifiable k-time and epoch-based redactable blockchain (KERB), which uses a financial penalty to control rewriting rights and punish malicious behavior. Shen et al. [26] developed a redactable blockchain with verification built on a double-trapdoor chameleon hash function, allowing efficient and key-exposure-resistant block editing. Li et al. [27] proposed the non-interactive chameleon hash (NITCH). This process issues the trapdoor key to nodes in the group, and the participants independently compute their portions without communicating.

TABLE I: Comparisons of state-of-art redactable blockchain schemes.

Scheme	Multi Authority	Multiple Block Redaction	Block Level Redaction	Transaction Level Redaction	Platform
Ateniese [15]	×	×	Yes	Yes	Bitcoin
Jia [16]	Yes	×	Yes	×	Proof-of-Concept Simulation
Deuber [20]	×	×	Yes	×	×
Xu [21]	Yes	×	Yes	×	Ethereum
Zhang [28]	Yes	×	Yes	×	Charm Library
Zhou [29]	Yes	×	Yes	Yes	Hyperledger Fabric + Golang
Our DACH	Yes	Yes	Yes	Yes	Hyperledger Fabric + Charm

Zhang et al. [28] proposed a redactable blockchain model where data sharing is secured with hierarchical access controls, using the chameleon hash and attribute-based encryption method. Data owners can determine which participants can modify their data when they set their access control policy, and modifications are authenticated with a digital signature. Zhou et al. [29] proposed the redactable blockchain, which utilizes a trapdoor hash for transaction editing, deletion, and smart contract repair. They also utilized holomorphic secret-sharing schemes to distribute the private key. Tian et al. [30] designed a blockchain authentication tree (BAT) that uses the chameleon hash function and aggregatable vector commitment to link blocks that are continuously added. However, their evaluation section lacks details regarding the performance of the redactable blockchain cryptographic functions.

Gu et al. [31] proposed a dependable and efficient decentralized trust management system based on consortium blockchain for intelligent transportation systems, where trust computation is performed on both vehicles and RSUs. Jia et al. [16] proposed a decentralized, traceable, and efficient redactable blockchain, which encodes redacted blocks in an RSA accumulator. Li et al. [32] proposed a redactable blockchain protocol that can be applied to proof-of-work and proof-of-stake blockchains and supports low-latency data redaction with high throughput.

A. Comparison with Existing Approaches

In this subsection, we compare our scheme with other state-of-the-art redactable blockchain systems based on the following features of each system: multi-authority support, multiple block redactions, block-level redaction, transaction-level redaction, and simulated blockchain platform. Specifically, we compare the proposed scheme with Ateniese's scheme [15], Jia's scheme [16], Deuber's scheme [20], Xu's scheme [21], Zhang's scheme [28], and Zhou's scheme [29].

As we observe from Table I, most of the proposed redactable blockchain schemes utilize the multi-authority setting, all supporting the block-level redaction. Only Ateniese's scheme [15], Zhou's scheme [29], and our proposed system provide transaction-level redaction, and our scheme is based on both chameleon hash and smart contracts-based operations. The redactable blockchain evaluation section provides detailed comparative experiments with Ateniese's scheme [15]. For Deuber's scheme [20], Xu's scheme [21], Zhang's scheme [28], Jia's scheme [16], and Zhou's scheme [29] we instead provide Table I to compare the proposed schemes, as their evaluations are based on different cryptography techniques, and utilize other metrics. Our proposed DACH scheme can solve the centralized authority issue and support multiple blocks, block-level, and transaction-level redaction.

IV. System Architecture

In this section, we describe the redactable blockchain architecture for vehicular event data recording applications. To enable trustful liability determination in CAV accident scenarios, and also address the challenges of data integrity, verification, accessibility, and modification of CAV accident forensics reports, this paper proposes a vehicular event forensics system based on a redactable permissioned blockchain. The proposed system enables multiple decentralized authorities to reach consensus on both block-level and transaction-level data redactions, while providing a transparent and auditable system for storing, retrieving, and modifying CAV accident data.

As shown in Fig. 1, we first describe listed entities.

- Permission issuer: The permission issuers, also referred to as the attribute-based authorities, include organizations such as the Department of Motor Vehicles (DMV), the Federal Highway Administration (FHWA), and the Insurance Company. They can execute redactable operations in a controlled and auditable manner if they reach consensus with the other stakeholders according to the threshold-based access policy.
- Cellular communication: Cellular communication refers to one of the communication networks for vehicles and infrastructure in ITS. In our scenario, vehicles can use a cellular network-based infrastructure to define an ad-hoc 'vehicular network' where all the vehicles that are served by the same base station of the vehicle(s) directly involved in the accident belong to the vehicular network.
- DSRC communication: DSRC, or Dedicated Short-Range Communications, is a wireless technology that enables vehicles to communicate with each other and with roadside infrastructure like traffic signals. In our scenario, accident vehicles and witness vehicles use the IEEE 802.11p standard of DSRC to send and receive 'block generation' requests, which can be forwarded to the peer nodes and permission issuers by the RSU.
- RSU: A Roadside Unit (RSU) is a device used in ITS and connected vehicle environments that serves as a communication gateway between the vehicles, infrastructure, and other networks. In our scenario, it can use the DSRC or cellular communication to send information, depending on their location relative to the accident vehicles. The role can be the peer node.

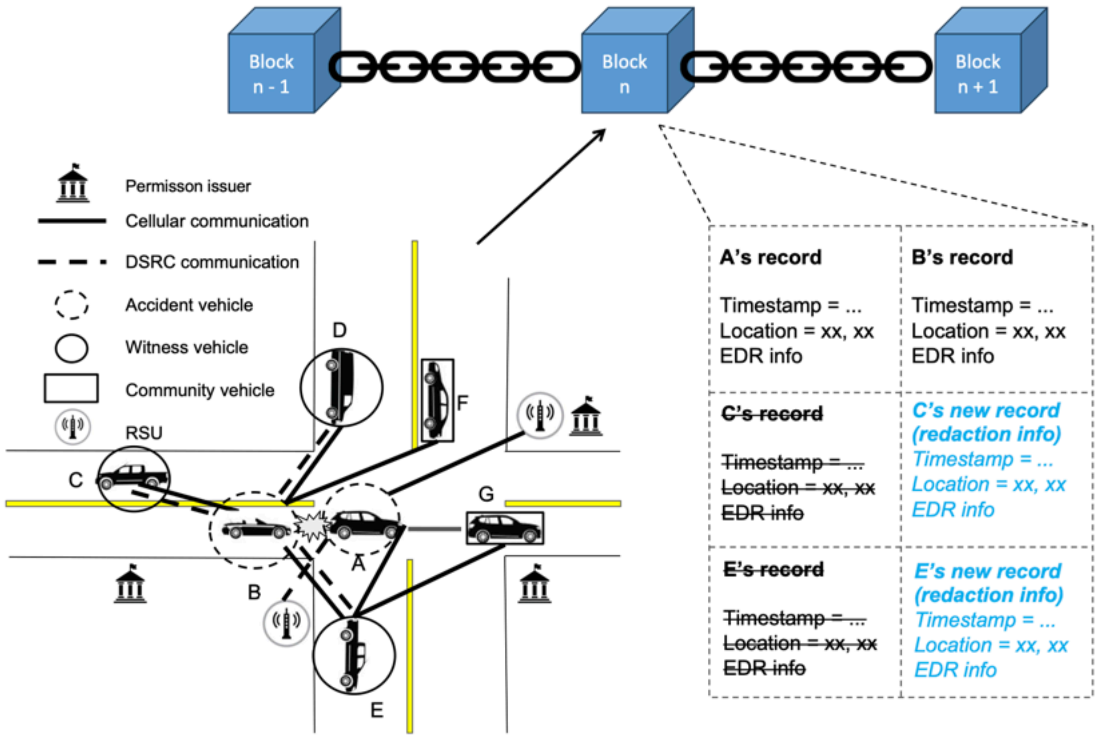


Fig. 1: Redactable blockchain for vehicular event forensics system.

- Accident vehicle: Vehicles directly involved in an accident are termed "accident" vehicles. In the blockchain network, the role is the client node.
- Witness vehicle: vehicles within the DSRC transmission range from the accident scene are termed "witness" vehicles. The role can be the client node.
- Community vehicle: vehicles within the vehicular network but outside the DSRC transmission range from the accident scene are termed "community" vehicles. The role is peer node.

Next, we define three levels of blockchain redaction operations.

1) *Multiple block redaction:* Multiple blocks could be redacted and replaced by one new block.
2) *Block-level redaction:* One block could be replaced entirely by one new block.
3) *Transaction-level redaction:* Merkle tree data could be replaced by the same hash value within one or multiple blocks.

A. Permissioned Blockchain Network

In the proposed system, a permissioned blockchain network provides a secure distributed ledger for saving accident events. Unlike permissionless blockchains, which allow anyone to participate in the network and perform operations, permissioned blockchains restrict read and write access to a select group of participants. This controlled access ensures higher security and integrity of the blockchain records. The permissioned blockchain network serves as a secure distributed ledger, enabling the transparent and immutable storage of accident events. Each accident event, including relevant data such as timestamps, vehicle information, and location, is recorded as a transaction on the blockchain. The distributed nature of the blockchain ensures that the accident events are replicated and stored across multiple nodes, enhancing data availability and resilience against tampering or loss.

Compared to permissionless blockchains, a permissioned blockchain network is better suited for supporting redactable operations in the context of a vehicular event forensics system. This restriction prevents malicious actors or unauthorized entities from tampering with or accessing sensitive accident event data. Additionally, the permissioned nature of the network facilitates compliance with regulatory requirements, privacy considerations, and data protection policies.

The permission issuers, also referred to as the attribute-based authorities, play a crucial role in conducting redactable operations for blockchain records. In the proposed design, the authorities, such as the Department of Motor Vehicles (DMV), the Federal Highway Administration (FHWA), the Insurance Company, Vehicle Manufacturer, and the National Highway Traffic Safety Administration (NHTSA), assume the responsibility of deploying and maintaining the permission issuers within the network in a decentralized way. Redactable operations involve selectively removing or modifying sensitive information from accident event records, such as personally identifiable information (PII) or classified details. By having designated permission issuers, the system ensures that multiple attribute-based authorities are responsible for executing redactable operations in a controlled and auditable manner (threshold-based access policy).

The controlled access to the network ensures that only authorized entities, such as "accident" vehicles, "witness" vehicles, and nearby roadside units, can participate as client nodes. The system achieves a decentralized architecture that combines the benefits of distributed ledger technology with controlled access and redactable operations. This design choice enhances the transparency, reliability, and security of the vehicular event forensics system while ensuring compliance with privacy and data protection regulations.

Peer nodes ("community" vehicles in our scenario) in a permissioned blockchain refer to the network participants that maintain a copy of the distributed ledger, validate transactions, and contribute to the consensus process. These peer nodes serve as the foundational building blocks of the blockchain infrastructure, contributing to transaction validation and overall network performance.

The new block generation and block redaction procedure is described as follows. As depicted in Fig. 1, the client nodes ("accident vehicles") will trigger the block generation operation. The witness group formation procedure (Details in the next subsection) is invoked when there is an event generation/redaction request, and then the system executes the block generation/redaction operation. Participants in the witness group will be notified through the broadcast and DSRC communication between themselves and all "accident" vehicles. After the "accident" and "witness" vehicles generate and broadcast accident event data into the vehicular network. Meanwhile, "community" vehicles are responsible for validating the received event data and confirming it with the lead community's vehicle. The lead "community" vehicle can execute the $t-of-n$ threshold voting scheme to achieve consensus when t out of n "community" vehicles confirm the accident event. We argue that the RSU is usually trustworthy and honest. If the accident location has a nearby RSU, the RSU could serve as the lead "community" node.

Later, if the "accident" and "witness" vehicles find that event data needs to be updated, then the redaction request will be triggered. The permission issuers (attribute-based authorities, such as the DMV and FHWA) will check the attribute-based threshold access policies. A legitimate redaction requires 't-of-n' attribute authorities to approve (e.g., 3-of-5 legal attribute-based policies). If the conditions have been satisfied, the redaction operation can be executed, and later new data can be recorded in the redacted block.

Overall, using a permissioned blockchain network in the proposed system provides a robust foundation for storing accident events, enabling redactable operations, and ensuring the integrity and security of the forensics data.

B. Witness Group Formation

In this subsection, we discuss the motivation and design of the location-based, witness group formation process in the proposed system. A location-based witness group formation that dynamically groups a controlled number, M, of reliable local nodes for the trust event generation processing is an essential process for lowering the computational complexity of event recording/redactable operation in our location-specific event forensics system. The witness group formation procedure is invoked when there is an event generation/redaction request, and then the system executes the block generation/redaction operation. We propose a trusted reputation mechanism to control group formation in our system to engage only the reliable witness nodes (near the accident vehicles) and safeguard blockchain system integrity. In the reputation mechanism, witness nodes participate by submitting transactions in client role, leading to the legitimate block generation or block redaction procedure, which will increase their reputation. Conversely, the reputation score of malicious nodes, such as providing unreliable or fake data to mislead the system, will be lowered if a redaction is later needed to correct the vehicle's error.

We propose the *reputation graph* $(\mathcal{R}, G)$ to formulate the reputation of witness nodes, where $\mathcal{R}$ and G are sets of nodes and arcs, respectively. The weight $\omega(\mathcal{R}_i, \mathcal{R}_j) \in [0, W]$ associated with arc $(\mathcal{R}_i, \mathcal{R}_j)$ represents the *reputation* where $\mathcal{R}_i$ assigns to $\mathcal{R}_j$, based on $\mathcal{R}_i$'s record of $\mathcal{R}_j$'s previous performance. Note that *reputation graph* does not have the self-arcs (i.e., $\nexists\ (\mathcal{R}_i, \mathcal{R}_i) \in G$), which indicates nodes cannot assign a reputation score to themselves. In our case, j is a witness candidate node and i is from a mix set $\mathcal{R}$ of nearby vehicles $\mathcal{R}_1$ and roadside units $\mathcal{R}_2$, $\mathcal{R}_j \in \mathcal{R}, \forall j \neq i$. $\mathcal{R}_1$ and $\mathcal{R}_2$ together form the $\mathcal{R}$ set.

We define the *normalized reputation* that $\mathcal{R}_i$ assigns to $\mathcal{R}_j$, $\rho(\mathcal{R}_i, \mathcal{R}_j) \in$ [0,1], by dividing $\omega(\mathcal{R}_i, \mathcal{R}_j)$ to sum of all reputation scores that $\mathcal{R}_i$ assigns as follows:

$$\rho(R_i, \mathcal{R}_j) = \frac{\omega(\mathcal{R}_i, \mathcal{R}_j)}{\sum_{\mathcal{R}_k \in \mathcal{I}(\mathcal{R}_i)} \omega(\mathcal{R}_i, \mathcal{R}_k)}, i \in \mathcal{R}_1. \tag{1}$$

$$\rho(R_{i'}, \mathcal{R}_j) = \frac{\omega(\mathcal{R}_{i'}, \mathcal{R}_j)}{\sum_{\mathcal{R}_k \in \mathcal{I}(\mathcal{R}_{i'})} \omega(\mathcal{R}_{i'}, \mathcal{R}_k)}, i' \in \mathcal{R}_2. \tag{2}$$

where $\mathcal{I}(\mathcal{R}_i) = \{R_j | \exists (\mathcal{R}_i, \mathcal{R}_j) \in G\}$ is the set of nodes where $\mathcal{R}_i$ has interacted with in the past. Note the summation over all normalized reputation scores that $\mathcal{R}_i$ and $\mathcal{R}_{i'}$ assign is one:

$$\sum_{\mathcal{R}_j \in \mathcal{R}, \forall j \neq i} \rho(\mathcal{R}_i, \mathcal{R}_j) + \sum_{\mathcal{R}_j \in \mathcal{R}, \forall j \neq i'} \rho(\mathcal{R}_{i'}, \mathcal{R}_j) = 1. \tag{3}$$

The reputation score of each witness node τ_j is computed by summation over all normalized reputation scores, which are given to node $\mathcal{R}_j$ as follows:

$$\tau_j = \sum_{\mathcal{R}_i \in \mathcal{R}, \forall i \neq j} \rho(\mathcal{R}_i, \mathcal{R}_j) + \sum_{\mathcal{R}'_i \in \mathcal{R}, \forall i' \neq j} \rho(\mathcal{R}'_i, \mathcal{R}_j). \tag{4}$$

Each witness node will store its own reputation score on the blockchain. The reputation scores of witness nodes will be updated according to new reputation scores assigned by the witness group involved in the transaction submission for block generation/redactable operation. The reputation score will remain the same if the witness node is not involved in the transaction submission for block generation/redactable operation.

We define the distance score of each node σ_j as follows:

$$\sigma_j = \frac{\sum_{\mathcal{R}_j \in \mathcal{R}} d_{\mathcal{R}_j}}{|\mathcal{R}| \times d_{\mathcal{R}_j}}, \tag{5}$$

where $d_{\mathcal{R}_j}$ is the distance of witness candidate node $\mathcal{R}_j$ from event forensics place, which is updated based on the location information at the time of invoking the process, and $|\mathcal{R}|$ is the size of set $\mathcal{R}$, the total number of nodes.

We formulate the witness group formation (WGF) as an Integer Program (IP), named IP-WGF. We also propose the following decision variables:

$$X_{\mathcal{R}_j} = \begin{cases} 1 & \text{if node } j \text{ is assigned to the witness group,} \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

We generate the IP-WGF as follows:

$$\text{Maximize} \sum_{\forall j} (\alpha \tau_j + \beta \sigma_j) X_{\mathcal{R}_j}, \tag{7}$$

Subject to:

$$\sum_{\mathcal{R}_j \in \mathcal{R}} X_{\mathcal{R}_j} = M, \tag{8}$$

$$X_{\mathcal{R}_j} \in \{0, 1\}. \tag{9}$$

The objective function (7) can maximize the combination of the witness group's reputation score and their geographical closeness. The objective function has non-negative coefficients, α and β, allowing the system to assign different weights to the reputation and distance based on event recording and forensics scenarios. Constraint (8) ensures that IP-WGF can select a pre-specified number of nodes, M, to be included in the witness group. M represents the number of nodes in the redactable operation. Constraints (9) indicate that decision variables are binary values.

C. Decentralized Attribute-based Chameleon Hash Operation

The existing chameleon hash functions and their variants are usually designed for the centralized setting, in which there is an entity that possesses the private key for redaction. However, in the blockchain system, it is hard to specify such an entity due to the requirement of decentralization. To address the centralized setting issue in traditional chameleon hash algorithms, we proposed the decentralized attribute-based chameleon hash (DACH) scheme. We first describe the decentralized attribute-based chameleon hash syntax.

Syntax: We utilize the $Out \leftarrow Algorithm\ in$ to represent the protocol executed by p participants $P_1, ..., P_t$, where the input and output of P_i are in_i and out_i. The decentralized attribute-based chameleon hash function $DACH = (AuthoritySetup, AuthorityKeyGeneration, DACHKeyGen, Hash, Collision, Verify)$ includes the following six protocols and algorithms.

Authority Setup$(IP, AT_i) \longrightarrow PK_i, SK_i$. Authority i takes the initial parameter IP and attributes AT_i (For example, the AT_i can represent the attribute of vehicle identity) as input value to generate public key pairs PK_i and secret master key SK_i.

Authority Key Generation$(GID, AT_i, SK_i, IP) \longrightarrow SK_i, _{GID}$. It takes as inputs attribute AT_i, global identification number GID of the participant p, the secret master key SK_i, and the initial parameter IP, and finally outputs attribute key $SK_i, _{GID}$ to the participants p.

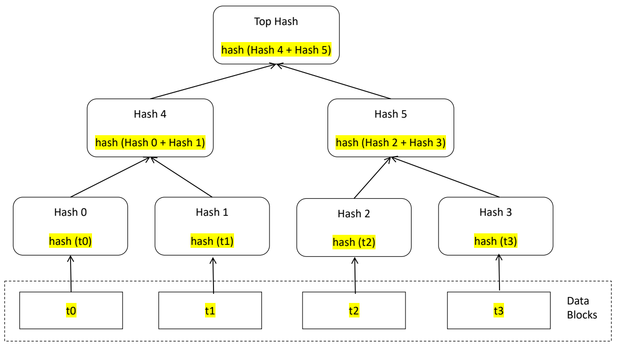


Fig. 2: Merkle Tree Structure

$pk_i, sk_i \leftarrow DACHKeyGen(1^\lambda, 1^p)$. In this key generation phase, each participant P_i inputs the security parameter λ and the number of participants p. This algorithm outputs the attribute-based public key pk_i and private key sk_i share. Our construction followed the threshold secret-sharing scheme, and the authorized sets are all sets whose size is bigger than some threshold, that is, they realize the t-out-of-n access structure $\mathbb{A}_t = \mathbb{A} \subseteq p_1, ..., p_n : |\mathbb{A}| \geq t$, where $1 \leq t \leq n$ is an integer [33].

$(h) \longleftarrow Hash(pk_i, m, r)$. This hashing algorithm takes as input a public key pk_i, the original message m, and the random number r. It will output the hash value h.

$\{r'\}_{i=1}^p \longleftarrow Collision(\{sk_i, h, (m, r), m'\}_{i=1}^p)$. In this redaction algorithm, the inputs are each participant's (attribute-based) share private key sk_i, hash value h, random message (m, r), and redacted message m', it outputs the redacted random number r'. The $Collision$ operation can be settled as a limited number, depending on the reality scenario (limited number of redaction requests).

$\{0, 1\} \longleftarrow Verify(pk_i, m, h, r) == Verify(pk_i, m', h, r')$. The verification algorithm takes the input as public key pk_i, the original message, hash value, and random number (m, h, r). The redaction message, hash value, and the new random number (m', h, r'), and it outputs the binary value, which indicates whether the redaction message (m', r') is valid or not.

With the input of the public key g^p, the original random message $(m, r = (g^e, g^{pe}))$, and the redacted message $(m', r' = (g^{e'}, g^{pe'}))$, the verification algorithm will compute $b \longleftarrow Hash(g^p, m, h, r)$ (b is the binary value). Finally, it outputs 1 if $Hash(g^p, m, h, r) = Hash(g^p, m', h, r')$, and 0 otherwise.

Finally, the correctness of the decentralized attribute-based chameleon hash scheme requires that $pk_i, sk_i \leftarrow DACHKeyGen(1^\lambda, 1^p)$, all $(h) \longleftarrow Hash(pk_i, m, r)$, and that $\{r'\}_{i=1}^p \longleftarrow Collision(\{sk_i, h, (m, r), m'\}_{i=1}^p)$. We have that $Hash(pk_i, (m, r)) = h$ and that $Verify(pk_i, m, h, r) == Verify(pk_i, m', h, r')$.

D. Redactable Algorithms

1) Merkle Tree Basics: The transactions in a block are not stored as plain text, but rather in a data structure known as the Merkle tree. A Merkle tree is the data structure utilized in blockchain technology to organize and verify transactions within a block. Using hash functions like MD5 [34], BLAKE2 [35], SHA-1 [36], or SHA-256 [37], each transaction is individually hashed to create a unique fingerprint. These hashed transactions are then combined in pairs and hashed again, continuing this process until a single hash, known as the Merkle root, is obtained (Fig. 2). The Merkle root is then stored in the block header, serving as an efficient representation and validation of the integrity of all the transactions within the block.

2) Multiple Block Redaction: We first introduce the multiple block redaction operation. To enable this functionality, we employ the decentralized attribute-based chameleon hash function. Unlike the classical chameleon hash functions, the decentralized attribute-based chameleon hash functions allow each participant to own its shared secret key sk_i without a central authority.

Utilizing the decentralized attribute-based chameleon hash function, our blockchain enables multiple blocks to be redacted. This innovative approach allows for transactions to be edited in multiple, consecutive blocks while ensuring consistency between the original and previous block and transaction hash values, maintaining the cryptographic integrity of the blockchain data after a redaction is processed.

As shown in Algorithm 1, the function of a multiple-block redaction generation is executed through the blockchain network when multiple attribute authority nodes have received the redaction request. This algorithm builds the $Block_{new}$ and $Rand_{new}$ from the existing blockchain system. Lines 1-3 present the preparation for redacting the new block. Lines 4-8 present the attribute-based conditions if it satisfies the access policies $(DMV.driverid == xx) \wedge (range.start\_block == xx) \wedge (range.end\_block == xx)$. Line 9-13 presents the redaction operation if it satisfies the predefined condition $p\_hash_i == l\_hash_{i+n}$ && $l\_hash_i == p\_hash_{i-n}$ and the $H(pk_i||B_xid'||\mathbb{A}) = h_{B-DACH})$, where p_hash indicates the header's hash value of the preceding block, and l_hash stores the header's hash value of the block that is redacted to the current block. B_xid' is the newly redacted block, $\mathbb{A}$ is the access policy defined before, and h_{B-DACH} is the newly generated block-level hash value. Line 14 finally outputs the $Block_{new}$ and $Rand_{new}$ for a new redacted block. The detail of the multiple block redaction workflow is shown in Fig. 3.

3) Transaction-level Redaction: Our blockchain implementation introduces a unique feature that also supports redactions at the transaction level. We employ a modified version of the Merkle tree structure that utilizes the decentralized chameleon hash function to enable this functionality. Unlike traditional hash functions, decentralized chameleon hash functions possess the remarkable property of allowing transaction modifications without altering the corresponding hash values. A special transaction-level trapdoor key is required for this operation.

Algorithm 1: Multiple Block Redaction Operation

```
Input  : $n$ multiple block redactions have been
         received for the redaction operation
Output: $Block_{new}$ and $Rand_{new}$ for the new redacted
         block
1  while There exists a new block redaction request has
     been received for the blockchain do
2      Execute multiple block redaction operation;
3  end
4  if (DMV.driverid == xx) $\wedge$ (range.start_block == xx)
     $\wedge$ (range.end_block == xx) then
5      Pass the attribute-based access policy;
6  else
7      Deny the redaction request;
8  end
9  if
     $p\_hash_i == l\_hash_{i+n}$ && $l\_hash_i == p\_hash_{i-n}$
     && $H(pk_i||B_xid'||\mathbb{A}) = h_{B-DACH})$ then
10     redact the new block;
11 else
12     wait for another round of redaction operation;
13 end
14 The system updates $Block_{new}$ and $Rand_{new}$ for new
     redacted block;
```

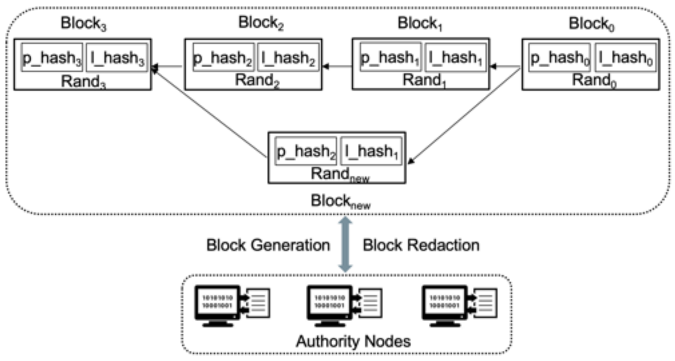


Fig. 3: Multiple block redaction workflow.

Utilizing the decentralized chameleon hash function, our blockchain enables the redaction of specific transactions within a block without necessitating changes to the associated hash values. This innovative approach allows for transaction editing while preserving the original hash values, ensuring the integrity and consistency of the Merkle tree structure. As shown in Algo. 2, Lines 1-3 present the preparation for redacting the transaction. Lines 4-8 present the attribute-based conditions if it satisfies the access policies $(DMV.driverid == xx) \wedge (transaction.id == xx)$. Lines 9-13 present the redaction operation if it satisfies the predefined condition $hash(t_i) == hash(t_i')$ and the $H(pk_i||id||t_xid'||\mathbb{A}) = h_{T-DACH})$, where t_xid' is the newly redacted transaction, A is the prior mentioned access policy, and h_{T-DACH} is the newly generated transaction-level hash value. Line 14 outputs the new transaction t_i' while preserving the original hash values.

Algorithm 2: Transaction Level Redaction Operation

```
Input  : Transaction $t_i$ has been received for the
         redaction operation
Output: New redacted transaction $t_i'$
1  while There exists a new transaction redaction request
     has been received for the blockchain do
2      Execute transaction redaction operation;
3  end
4  if (DMV.driverid == xx) $\wedge$ (transaction.id == xx) then
5      Pass the attribute-based access policy;
6  else
7      Deny the redaction request;
8  end
9  if $hash(t_i) == hash(t_i')$ &&
     $H(pk_i||id||t_x id'||\mathbb{A}) = h_{T-DACH}$) then
10     Redact the new transaction;
11 else
12     Wait for another round of redaction operation;
13 end
14 The system updates the transaction $t_i$ into new
     redacted transaction $t_i'$;
```

V. Security Analysis

We first construct the correctness of the decentralized attribute-based chameleon hash functions; given any m_1, m_2, and r_1, how can we find that $r_2 \in Z_p^*$ which satisfy $Hash\ (m_1, r_1) = Hash\ (m_2, r_2)$.

Proof. Suppose that p and q are two large prime numbers, Z_p^* is the group of order q, and g is the generator of Z_p^*. Let trapdoor sk be $x \in Z_p^*$, then the public key pk denotes as $y = g^x\ mod\ p$. Given the message $m_1 \in Z_p^*$ and the random number $r_1 \in Z_p^*$, then the chameleon hash value of m_1 is represented as $Hash(m_1, r_1) = g^{m1} y^{r1}\ mod\ p$.

Since

$$Hash(m_1, r_1) = g^{m_1} y^{r_1}\ mod\ p = g^{m_1 + xr_1}\ mod\ p,$$

$$Hash(m_2, r_2) = g^{m_2} y^{r_2}\ mod\ p = g^{m_2 + xr_2}\ mod\ p.$$

Then we have:

$$r_2 = \frac{m_1 - m_2}{x} + r_1\ mod\ p.$$

■

We examine the redactable blockchain concerning the design goals in Section IV through the following theorems.

Theorem 1 (Secure): **The proposed DACH scheme is secure against the collusion attack.**

Proof. The proposed DACH system assigns every participating entity a globally unique identity $u = H_1(GID)$. In the worst case, if two or more entities collude on the attribute-based key component to access unauthorized information, their GID are different and cannot obtain a valid attribute key. If the redaction access policy is encoded over attribute sets $\mathcal{S}$, after the collusion, they may obtain the set of attributes to pass the access policy. However, it is obvious that each component C_i belonging to $SK_{i,GID}$ is different; therefore, the trapdoor secret key remains invalid and blind. As a result, the proposed solution is secure against collusion attacks among multiple participating entities. ■

Theorem 2 (Editable): **When the editor with the right to edit receives at least n sub-keys and correctly recovers the attribute-based private key, the editor edits and ensures hash consistency.**

Proof. When the editor receives at least n attribute-based private keys S_i, it uses the Lagrange interpolation functions to recover the trapdoor private key sk (with permission or limited time).

The decentralized attribute-based chameleon hash function $DACH(m, r, h) = g^m h^r\ mod\ p$, and public key is $h = g^s$, and sk is the private key, m is the message, and r is the random number. When we have $DACH(m, r, h) = DACH(m', r', h')$, $g^m h^r\ mod\ p = g^{m'} h^{r'}\ mod\ p$, then we get $m + Sr = m' + Sr'$. Lastly, $r' = m - m' + \frac{Sr}{S}\ mod\ p$.

Therefore, editors use a new random number r, which can ensure the hash consistency after editing. It will not affect the block before and after the hash value, so the proposed scheme is editable with trust.

■

Theorem 3 (Resistance): **A DACH blockchain system can effectively resist Sybil attacks if the underlying chameleon hash function is collision-resistant and the attribute verification process is robust.**

Proof. In a Sybil attack, the attacker needs to generate a large number of valid secret keys (sk) to gain control of the network. To generate a valid secret key (sk), the attacker needs to find a valid initial parameter IP and a new attributes token AT' for the chameleon hash function. Due to the collision-resistance of the chameleon hash function, finding a new attributes token AT' that collides with an existing one AT is computationally infeasible without the valid global identification number GID and the secret master key SK_i.

If the attacker could generate multiple valid identities, it indicates that they can create multiple valid unauthorized attribute tokens. This is equivalent to finding collisions in the chameleon hash function. Since the chameleon hash is collision-resistant, this is computationally infeasible for a single attacker without the valid secret key (sk). The decentralized blockchain ensures that the verification process is distributed, preventing a single point of failure in identity validation. The blockchain verifies the authenticity of attributes by checking against a distributed ledger of verified attributes or using cryptographic proofs. Therefore, a DACH blockchain system can effectively resist Sybil attacks because it makes it computationally infeasible for attackers to generate a large number of valid identities. ■

VI. Experiments and Evaluations

A. Performance of Blockchain Network

1) Setup: The Hyperledger Fabric blockchain v2.2 open-source software was used to create the blockchain module. The

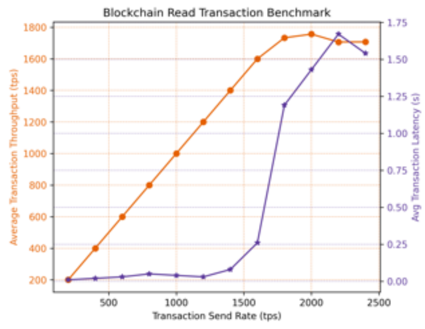


Fig. 4: Transaction Latency and Average Transaction Throughput as a function of Transaction Send Rate for the READ operation.

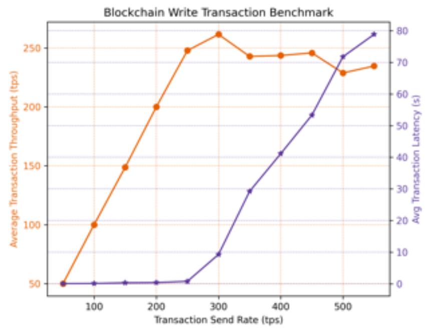


Fig. 5: Transaction Latency and Average Transaction Throughput as a function of Transaction Send Rate for the WRITE operation.

network was tested using a virtual machine having 8 cores of an Intel i9-10900K processor at 3.70 GHz and 24GB of RAM. We set up four peers, five orderers, and two organizations using the Raft consensus algorithm. In the experiments, the Hyperledger Caliper tool was utilized to measure the transaction throughput and latency of the blockchain network to quantify its performance [38], [39].

The capacity of a blockchain network to process transactions is measured by its average transaction throughput, which is the average number of completed transactions per second (tps) processed by the blockchain peers and orderers during a testing cycle. Latency is the average time it takes for a transaction to be constructed by the clients and then successfully committed to the ledger. To assess these metrics, we increased the rate of input transactions per second sent by blockchain clients. In other words, the transaction send rate represents the number of transactions per second input to the blockchain network by the clients for the duration of a testing cycle. Two transaction types are tested, notably read and write transactions, which correspond to retrieving data from and storing data on the blockchain network, respectively.

2) Read Transaction: In Fig. 4, we illustrate our blockchain network's average transaction throughput and average transaction latency for the read operation. We conducted multiple experiments at different transaction send rates, and the results indicate that the blockchain network performance remains stable until the send rate reaches 1800 tps. A performance bottleneck occurs when the tps exceeds 1800, and the average transaction throughput stabilizes around 1750 tps. For the average transaction latency, the finalization time for the client to receive a response for a data retrieval request was on the order of milliseconds when the send rate was below 1800 tps, but quickly increased for higher send rates. While Hyperledger Fabric's execute-order-validate transaction model allows peer nodes to execute transactions concurrently, the concurrency limit for endorsement is set to 2500 transactions by default.

3) Write Transaction: In Fig. 5, we illustrate the average transaction throughput and the average transaction latency of our blockchain network for the write operation. Similarly to the read operation, a bottleneck is observed as the send rate increases, but the write transaction bottleneck happens at a much lower send rate, between 300 and 350 tps. The average transaction latency increases sharply as the send rate exceeds 300, but at a much greater magnitude than the read operation. For input send rates below 300 tps, the average write transaction finalization time was also on the order of milliseconds, but quickly reached a peak of 78.81 seconds at a send rate of 550 tps. Writing new data to the blockchain is more expensive for the peer nodes than data retrieval, resulting in a backlog of pending transactions and degraded performance.

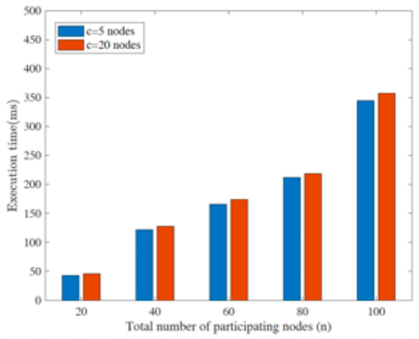
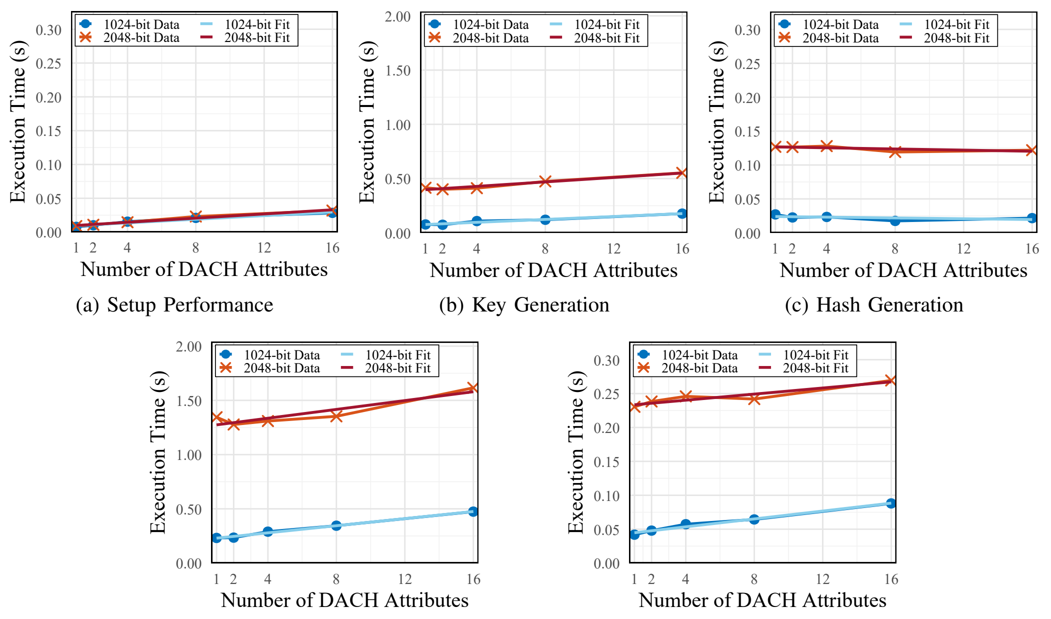


Fig. 6: Witness group formation cost vs. the total number of participating nodes.

B. *Witness Group Formation Cost*

We evaluate the changes in witness group size and the total number of participating nodes for the execution time of the witness group formation scheme. IP-WGF is optimally solved utilizing Python 3.0 libraries.

As we can observe from Fig. 6, we modified the total number of participating nodes from 20 to 100, and the witness's group (c) is configured as 5 and 20 nodes. The running times for 20, 40, 80, and 100 participating nodes are 43ms and 46ms, 122ms and 128ms, 166ms and 174ms, 212ms and 219ms, and 345ms and 357ms for $c = 5$ and $c = 20$ cases. When the total number of participating nodes increases, the running time to form the witness group also grows significantly. It indicates that witness group size has relatively less impact on execution time than the total number of participating nodes when c is relatively tiny. This is because the time complexity of the IP-WGF method is $O(n\ log(n))$, where n is the total number of participating nodes.



(a) Setup Performance (b) Key Generation (c) Hash Generation (d) Hash Collision (e) Hash Verification

Fig. 7: Execution time performance of DACH algorithm operations with varying numbers of attributes and different key sizes.

C. Performance of Decentralized Attribute-based Chameleon Hash

The Chameleon hash function was prototyped in Python using the Charm cryptographic library [40]. We conducted a performance evaluation of the proposed Decentralized Attribute-based Chameleon Hash (DACH), focusing on marginal cost analysis. Multiple measurements were averaged for each configuration to ensure statistical reliability. The assessment examined five core operations across different numbers of DACH attributes (1, 2, 4, 8, and 16) for both 1024-bit and 2048-bit key sizes, as shown in Fig. 7. For 1024-bit keys, prime numbers p and q each consisted of 309 random decimal digits, while for 2048-bit keys, they consisted of 617 random decimal digits. When the attribute count equals 1, DACH performance is equivalent to the Ateniese scheme [15] for all operations except setup, as the Ateniese scheme does not require attribute initialization, while our DACH algorithm includes this step to support multiple attributes.

1) Marginal Cost Analysis and Linear Regression: Marginal cost analysis quantifies the incremental performance impact of adding one additional attribute to an existing system. In economic terms, marginal cost represents the additional resource required to produce one more unit of output. For our DACH system, given N existing attributes, the marginal cost of introducing the (N+1)th attribute is defined as:

$$MC_{n\rightarrow n+1} = T(n+1) - T(n) \tag{10}$$

where T(n) represents the average execution time for n attributes. Analysis of our experimental data from the five performance figures revealed that both 1024-bit and 2048-bit data points demonstrate approximately linear relationships between attribute count and execution time across most operations. This observed linear trend in our dataset justifies the application of linear regression modeling. We apply linear regression:

$$T(n) = \beta_0 + \beta_1 \cdot n + \varepsilon \tag{11}$$

where β_1 represents the slope coefficient. By definition, in linear regression, β_1 measures the average change in the dependent variable (execution time) for each unit increase in the independent variable (attribute count). This mathematical definition precisely corresponds to our marginal cost concept, making β_1 the average marginal cost per attribute. Therefore, linear regression provides both a fitting method for our data and a direct quantification of marginal costs.

2) Performance Analysis and Results: Table II presents the linear regression results for all DACH operations, organized by operation type and ranked by coefficient of determination (R^2) within each category.

TABLE II: Linear Regression Analysis of DACH Performance

Operation	**Key Size**	**R^2**	**Marginal Cost (s/attr)**
Setup	1024-bit	0.953	0.0013
	2048-bit	0.978	0.0015
Key Generation	1024-bit	0.967	0.0067
	2048-bit	0.959	0.0102
Hash Generation	1024-bit	0.271	-0.0003
	2048-bit	0.476	-0.0004
Hash Collision	1024-bit	0.994	0.0164
	2048-bit	0.846	0.0204
Hash Verification	1024-bit	0.981	0.0029
	2048-bit	0.877	0.0022

Fig. 7a shows setup operation execution times below 0.03 seconds for 1024-bit and 0.04 seconds for 2048-bit operations at 16 attributes, with strong correlation ($R^2 > 0.95$) and marginal costs of 0.0013-0.0015 seconds per attribute. This low overhead reflects simple parameter initialization requiring basic memory allocation. Fig. 7b reveals key generation times ranging from 0.08 to 0.18 seconds for 1024-bit and 0.4 to 0.55 seconds for 2048-bit operations. Both show good linear correlation ($R^2 > 0.95$), with 2048-bit operations having higher marginal costs (0.0102 s/attr) compared to 1024-bit (0.0067 s/attr) due to generating large random prime numbers for each attribute's key pairs. Hash generation in Fig. 7c shows flat trends with average 1024-bit operations around 0.02 seconds and average 2048-bit operations around 0.12 seconds. Poor linear correlation ($R^2 < 0.5$) and near-zero marginal costs indicate hash computation is dominated by fixed algorithmic costs. Fig. 7d demonstrates the most intensive operation, with 1024-bit collision generation ranging from 0.23 to 0.47 seconds and 2048-bit from 1.34 to 1.62 seconds. Strong linear correlation ($R^2 > 0.8$) reflects the highest marginal costs due to string processing and collision algorithms requiring random number generation for each attribute. Hash verification in Fig. 7e remains below 0.3 seconds with minimal marginal costs below 0.003 seconds per attribute, as verification involves modular arithmetic on fixed-length hashed messages.

D. Scalability Discussion

The system performance is sufficient to accommodate a large number of autonomous vehicles, as the transaction volume generated by vehicular event forensics tasks remains manageable within the observed throughput limits. As shown in Fig. 4 and Fig. 5, the observed throughput of 1800 tps for read operations and approximately 300 tps for write operations demonstrates the system's capacity to support a large number of autonomous vehicles. Notably, read and write operations are not frequent in practice, as they are primarily triggered by accident recording or querying events.

As for the DACH function, the linear regression models with high R^2 values enable reliable performance prediction for larger attribute sets. Using marginal cost coefficients from Table II, we can calculate execution times for practical scenarios. Setup operations maintain minimal overhead with predicted times below 0.05 seconds even at 32 attributes, while key generation needs 0.28 and 0.71 seconds for 1024-bit and 2048-bit operations, respectively. Hash collision with 1024-bit keys requires 0.54 seconds for 20 attributes and 0.74 seconds for 32 attributes. Hash verification operations remain below 0.3 seconds even at 32 attributes. Based on the poor linear correlation observed above, hash generation reflects algorithm design with fixed computational costs regardless of attribute count, resulting in execution times approximating average values rather than linear scaling.

In practical applications, setup operations establish access control policies to ensure proper attribute authorization, followed by key generation that typically occurs only once during system initialization rather than repeatedly. Redactable operations are infrequent operations performed only by authorized attributes governed by predefined access policies, and attribute participation typically remains moderate. The marginal cost analysis, combined with the infrequent nature of redactable operations, further ensures scalability. The bounded marginal costs across all operations confirm that DACH maintains good scalability for real-world deployment.

VII. Conclusion

This paper proposes a redactable blockchain for vehicular event forensics systems. We propose the decentralized attribute-based chameleon hash scheme, where multiple authority nodes generate the public and private key shares. We also extend the decentralized attribute-based chameleon hash to support both the multiple block redaction and transaction-level redacting operations. We implement the proposed architecture using Hyperledger Fabric, and the Charm cryptography library. The experiment results indicate that the proposed scheme is secure and efficient for decentralized chameleon hash schemes and real-world vehicular event forensics tasks.

References

[1] M. M. Islam, A. A. R. Newaz, L. Song, B. Lartey, S.-C. Lin, W. Fan, A. Hajbabaie, M. A. Khan, A. Partovi, T. Phuapaiboon *et al.*, "Connected autonomous vehicles: State of practice," *Applied Stochastic Models in Business and Industry*, vol. 39, no. 5, pp. 684–700, 2023.

[2] X. Zhang, H. Sun, X. Pei, L. Guan, and Z. Wang, "Evolution of technology investment and development of robotaxi services," *Transportation Research Part E: Logistics and Transportation Review*, vol. 188, p. 103615, 2024.

[3] I. Suarez, "The future of transportation-emerging self-driving taxis," in *29th Annual Western Hemispheric Trade Conference April 9-11, 2025 Conference Proceedings*, 2025, p. 158.

[4] California Department of Transportation, "Autonomous Vehicle Collision Reports," https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/autonomous-vehicle-collision-reports/.

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

[6] S. Zeba, P. Suman, and K. Tyagi, "Types of blockchain," in *Distributed Computing to blockchain*. Elsevier, 2023, pp. 55–68.

[7] W. Xu, X. Li, Q. Xu, Y. Zhang, X. Wu, W. Li, R. Wang, G. Liang, and H. Guo, "A risc-v based soc for blockchain data integration in iot edge devices," *Microelectronics Journal*, vol. 161, p. 106697, 2025.

[8] G. Bendiab, A. Hameurlaine, G. Germanos, N. Kolokotronis, and S. Shiaeles, "Autonomous vehicles security: Challenges and solutions using blockchain and artificial intelligence," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3614–3637, 2023.

[9] R. Jabbar, E. Dhib, A. B. Said, M. Krichen, N. Fetais, E. Zaidan, and K. Barkaoui, "Blockchain technology for intelligent transportation systems: A systematic literature review," *IEEE Access*, vol. 10, pp. 20 995–21 031, 2022.

[10] C. Meese, H. Chen, W. Li, D. Lee, H. Guo, C.-C. Shen, and M. Nejad, "Adaptive traffic prediction at the its edge with online models and blockchain-based federated learning," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[11] D. Gautam, G. Thakur, P. Kumar, A. K. Das, and Y. Park, "Blockchain assisted intra-twin and inter-twin authentication scheme for vehicular digital twin system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 10, pp. 15 002–15 015, 2024.

[12] L. Huang, C. Ge, X. Mao, and S. Yu, "Darb: Decentralized, accountable and redactable blockchain for data management," *IEEE Transactions on Network and Service Management*, 2024.

[13] L. Pan, H. Guo, and W. Li, "C-pfl: A committee-based personalized federated learning framework," *Journal of Network and Computer Applications*, p. 104327, 2025.

[14] W. Liu, Z. Wan, J. Shao, and Y. Yu, "Hypermaze: Towards privacy-preserving and scalable permissioned blockchain," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 360–376, 2023.

[15] G. Ateniese, B. Magri, D. Venturi, and E. Andrade, "Redactable blockchain – or – rewriting history in bitcoin and friends," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017, pp. 111–126.

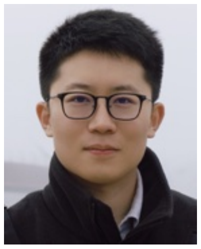
[16] M. Jia, J. Chen, K. He, R. Du, L. Zheng, M. Lai, D. Wang, and F. Liu, "Redactable blockchain from decentralized chameleon hash functions," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2771–2783, 2022.

[17] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*. Acm, 2006, pp. 89–98.

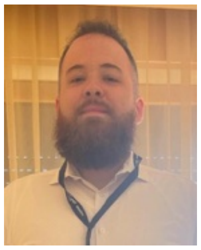
[18] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2011, pp. 568–588.

[19] H. Krawczyk and T. Rabin, “Chameleon hashing and signatures,” *Cryptology ePrint Archive*, 1998.
[20] D. Deuber, B. Magri, and S. A. K. Thyagarajan, “Redactable blockchain in the permissionless setting,” in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 124–138.
[21] Y. Xu, S. Xiao, H. Wang, C. Zhang, Z. Ni, W. Zhao, and G. Wang, “Redactable blockchain-based secure and accountable data management,” *IEEE Transactions on Network and Service Management*, pp. 1–1, 2023.
[22] D. Derler, S. Kai, D. Slamanig, and C. Striecks, “Fine-grained and controlled rewriting in blockchains: Chameleon-hashing gone attribute-based,” in *26th Annual Network and Distributed System Security Symposium, NDSS 2019*, 2019.
[23] Z. Zhang, T. Li, Z. Wang, and J. Liu, “Redactable transactions in consortium blockchain: Controlled by multi-authority cp-abe,” in *Australasian Conference on Information Security and Privacy*, 2021.
[24] K. Huang, Y. Mu, F. Rezaeibagha, X. Zhang, and X. Li, “Monero with multi-grained redaction,” *IEEE Transactions on Dependable and Secure Computing*, 2023.
[25] S. Xu, J. Ning, J. Ma, X. Huang, and R. H. Deng, “K-time modifiable and epoch-based redactable blockchain,” *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4507–4520, 2021.
[26] J. Shen, X. Chen, Z. Liu, and W. Susilo, “Verifiable and redactable blockchains with fully editing operations,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 3787–3802, 2023.
[27] J. Li, H. Ma, J. Wang, Z. Song, W. Xu, and R. Zhang, “Wolverine: A scalable and transaction-consistent redactable permissionless blockchain,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1653–1666, 2023.
[28] T. Zhang, L. Zhang, Q. Wu, Y. Mu, and F. Rezaeibagha, “Redactable blockchain-enabled hierarchical access control framework for data sharing in electronic medical records,” *IEEE Systems Journal*, vol. 17, no. 2, pp. 1962–1973, 2023.
[29] G. Zhou, X. Ding, H. Han, and A. Zhu, “Fine-grained redactable blockchain using trapdoor-hash,” *IEEE Internet of Things Journal*, 2023.
[30] G. Tian, J. Wei, M. Kutyłowski, W. Susilo, X. Huang, and X. Chen, “Vrbc: A verifiable redactable blockchain with efficient query and integrity auditing,” *IEEE Transactions on Computers*, vol. 72, no. 7, pp. 1928–1942, 2023.
[31] C. Gu, B. Ma, and D. Hu, “A dependable and efficient decentralized trust management system based on consortium blockchain for intelligent transportation systems,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2024.
[32] X.-Y. Li, J. Xu, L.-Y. Yin, Y. Lu, Q. Tang, and Z.-F. Zhang, “Escaping from consensus: Instantly redactable blockchain protocols in permissionless setting,” *IEEE Transactions on Dependable and Secure Computing*, pp. 1–20, 2022.
[33] A. Beimel, “Secret-sharing schemes: A survey,” in *International conference on coding and cryptology*. Springer, 2011, pp. 11–46.
[34] Z. Yong-Xia and Z. Ge, “Md5 research,” in *2010 second international conference on multimedia and information technology*, vol. 2. IEEE, 2010, pp. 271–273.
[35] J.-P. Aumasson, S. Neves, Z. Wilcox-O’Hearn, and C. Winnerlein, “Blake2: simpler, smaller, fast as md5,” in *Applied Cryptography and Network Security: 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings 11*. Springer, 2013, pp. 119–135.
[36] X. Wang, Y. L. Yin, and H. Yu, “Finding collisions in the full sha-1,” in *Advances in Cryptology–CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005. Proceedings 25*. Springer, 2005, pp. 17–36.
[37] H. Gilbert and H. Handschuh, “Security analysis of sha-256 and sisters,” in *International workshop on selected areas in cryptography*. Springer, 2003, pp. 175–193.
[38] H. Guo, W. Li, and M. Nejad, “A hierarchical and location-aware consensus protocol for iot-blockchain applications,” *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.
[39] H. Guo, C. Meese, W. Li, C.-C. Shen, and M. Nejad, “B2sfl: A bi-level blockchained architecture for secure federated learning-based traffic prediction,” *IEEE Transactions on Services Computing*, vol. 16, no. 6, pp. 4360–4374, 2023.
[40] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, “Charm: a framework for rapidly prototyping cryptosystems,” *Journal of Cryptographic Engineering*, vol. 3, pp. 111–128, 2013.

Hao Guo received the B.S. and M.S. degrees from the Northwest University, Xi’an, China in 2012, and the Illinois Institute of Technology, Chicago, United States in 2014, and his Ph.D. degree from the University of Delaware, Newark, United States in 2020, all in computer science. He is currently an Assistant Professor with the School of Software at the Northwestern Polytechnical University. His research interests include blockchain and distributed ledger technology, data privacy and security, cybersecurity, cryptography technology, and Internet of Things (IoT). He is a member of both ACM and IEEE.



Wanxin Li is a Assistant Professor in the Department of Communications and Networking at Xi’an Jiaotong-Liverpool University. He received his B.S. degree from Chongqing University in 2015, and his M.S. and Ph.D. degrees from the University of Delaware in 2017 and 2022, respectively. He was a recipient of IEEE ITSS Best Dissertation Award and IEEE TEMS Outstanding Ph.D. Dissertation Award in 2022. His research interests include blockchain, cryptography, and federated learning. He is a member of IEEE and ACM.



Collin Meese received his B.S. degree in computer science from the University of Delaware in 2020. He is currently working toward the Ph.D. degree at the University of Delaware. His research interests include blockchain, vehicular networks, distributed and high-performance computing, connected and autonomous vehicles, and intelligent civil systems. He is a student member of IEEE.

Yetong Wang received his B.S. degree from Xi’an Jiaotong-Liverpool University in 2023 and his M.S. degree from The Australian National University, Canberra, Australia in 2025. He is currently working toward the Ph.D. degree at Xi’an Jiaotong-Liverpool University. His research interests include blockchain, cryptography, and artificial intelligence.

Mark Nejad is an Assistant Professor at the University of Delaware. His research interests include network optimization, distributed systems, blockchain, game theory, and automated vehicles. He has published more than fifty peer-reviewed papers and received several publication awards including the best doctoral dissertation award of the Institute of Industrial and Systems Engineers (IISE) and the CAVS best paper award from IEEE VTS. His research is funded by the National Science Foundation and the Department of Transportation. He is a member of INFORMS and a senior member of IEEE.