Privacy-Preserving and Gas-Efficient Car Service Record Management Using Adaptive User-Resilient CKKS Homomorphic Encryption and Blockchain

S. Yasmin
Department of Computer Applications, B.S.Abdur Rahman Crescent Institute of Science & Technology, Chennai, India
yasmin_ca_jan21@crescent.education

G. Shree Devi
Department of Computer Applications, B.S.Abdur Rahman Crescent Institute of Science & Technology, Chennai, India
shreedevi@crescent.education (corresponding author)

Received: 13 October 2025 | Revised: 10 November 2025 | Accepted: 21 November 2025

 DOI: https://doi.org/10.48084/etasr.15498

ABSTRACT

Many car-selling and car-service websites have sprung up as a result of the automobile industry's quick digitization, offering consumers convenient access to vehicle details and transaction histories. Due to privacy and data management restrictions, these platforms typically show only the most basic information about an automobile, including its make, model, year of manufacture, mileage, price, and restricted ownership history, with Important service records, warranty claims, and part replacements typically left out. In addition to decreasing buyer transparency, the lack of thorough records makes it more difficult for automobile owners, dealers, and repair facilities to securely share data. Furthermore, data stored on centralized web servers is susceptible to loss of integrity, illegal changes, and data breaches. To overcome these obstacles, this study presents a unique method for the safe maintenance of comprehensive auto service records that combines blockchain technology with the AURA-CKKS homomorphic encryption scheme and the Brotli compression algorithm for storage optimization. The blockchain holds metadata, transaction logs, and Content Identifiers (CIDs) for traceability, while the InterPlanetary File System (IPFS) houses the compressed and encrypted data for decentralized and impenetrable storage. A Token-based Key Management System (TKMS) is utilized to securely distribute and maintain encryption keys among authorized stakeholders. The proposed architecture uses blockchain technology to ensure data immutability, AURA-CKKS encryption to maintain secrecy, and Brotli compression to reduce storage overhead, allowing all stakeholders to access verifiable and impenetrable vehicle histories while protecting data privacy. According to experimental results, the suggested framework decreases blockchain gas consumption by roughly 95% when compared to baseline raw on-chain storage, and by 94% compared to AURA-CKKS-based on-chain encryption. Brotli compression offers 43% reduction in data size. Additionally, in high-load circumstances, transaction confirmation time is decreased by 60%, query latency is reduced by 75%, and execution time is reduced by almost 55%. Due to adaptive bootstrapping and decentralized key verification, security testing reveals strong resilience (scores 4-5) against replay, Sybil, and key-theft attacks. These results confirm that the proposed architecture, combining AURA-CKKS, IPFS, TKMS, and blockchain, greatly improves computational efficiency, scalability, and privacy, providing a verifiable and tamper-proof vehicle-record management solution appropriate for next-generation automotive data ecosystems.

Keywords-homomorphic encryption; AURA-CKKS; blockchain; token-based key management; interplanetary file system

I. INTRODUCTION

The necessity of tamper-evident, privacy-preserving record-keeping while maintaining manageable storage and gas expenses has been highlighted by recent attempts to protect car-selling and service platforms. The CKKS scheme enables approximate arithmetic on encrypted real-valued data and is widely used for privacy-preserving analytics [1]. Since CKKS accumulates noise during homomorphic operations, several works have explored bootstrapping to refresh ciphertexts and support deeper computations, including schemes tailored for approximate homomorphic encryption [2-4]. Libraries such as Intel HEXL, SEAL, PALISADE, and HElib were used in implementation surveys [5-7]. By keeping big payloads off-chain and only maintaining metadata and CIDs on-chain, off-chain decentralized storage, such as IPFS, offers content-addressable storage that supports blockchain immutability [8]. Before encryption, high-ratio lossless compression (such as Brotli) significantly reduces the size of structured data, reducing storage and transaction costs [9].

In many blockchain-based car management systems, hashed pointers or raw records are stored directly on-chain, which raises gas prices and exposes private information [10, 11]. Since secure distribution and revocation are necessary while private keys remain off-chain, key management is still crucial. In [12], a token-based TKMS was compatible with threshold-based key lifetime models that are appropriate for multi-stakeholder settings.

TABLE I. COMPARISON OF STATE-OF-THE-ART PRIVACY-PRESERVING TECHNIQUES WITH THE PROPOSED AURA-CKKS FRAMEWORK

Ref.	Encryption/Approach	Strength	Limitations	Gap addressed in this work
[13]	Quantum cryptography	High security and future-proofing against quantum attacks	Requires quantum hardware; high computational overhead	Scalable classical-crypto security is offered by AURA-CKKS without quantum.
[14]	Hybrid FHE and RSA	Enhanced confidentiality and several layers of protection	High latency due to multi-stage encryption	AURA-CKKS uses ab improved FHE pipeline with adaptive noise management.
[15]	Polynomial-based contextual encryption	Context-sensitive data security	Unsuitable for homomorphic processes or numerical calculations	Supports full homomorphic computation on encrypted data
[16]	Encryption and anonymization	Strong identity concealment and anonymity	Does not support encrypted computing; irreversible anonymization	Maintains both usability and computational capability on ciphertext
[17]	Multi-party computation	Suitable for distributed privacy-preserving tasks	Requires great coordination and coordinated multi-party engagement costs.	Works single-party or multi-party without coordination overhead

II. RESEARCH MOTIVATION AND CONTRIBUTION

Massive volumes of vehicle-related data, including VIN, service histories, repair records, and technician comments, have been generated as a result of the rapid digitalization of the automotive ecosystem, with the data dispersed among brands, service providers, and geographical areas. However, when sharing these data with other stakeholders, privacy, scalability, and interoperability remain significant obstacles. Although traditional blockchain models guarantee data integrity, they suffer from high gas costs and the possibility of sensitive information being exposed when entire records are put directly on the chain. However, noise generation and performance limitations in deep calculations limit the use of homomorphic encryption systems such as CKKS, which permit computations on encrypted data.

These restrictions result in some significant issues that need to be resolved in any workable system:

- Computability vs. Privacy: Although standard encryption guarantees confidentiality, it makes it impossible to compute on encrypted data.
- Scalability and Gas Costs: Keeping big datasets on-chain reduces efficiency and increases gas usage.
- Noise Growth: Deep homomorphic calculations are limited by noise buildup in CKKS encryption.
- Key Management: The risk of key disclosure or misuse is increased by centralized or static key handling.

Few studies have successfully integrated data privacy, scalability, and computational efficiency. Previous research has examined safe data management utilizing blockchain or homomorphic encryption separately. Their practical use in multi-party automobile systems is further constrained by the lack of adaptive bootstrapping, compression optimization, and secure key management.

This paper presents the AURA-CKKS framework, a hybrid architecture that combines token-based key management, decentralized storage, and adaptive homomorphic encryption to enable safe and economical vehicle record preservation and address these research gaps. The proposed method incorporates:

- Adaptive bootstrapping in conjunction with AURA-CKKS encryption to manage noise development during encrypted computation;
- Brotli compression to increase on-chain and off-chain efficiency by reducing the size of ciphertext and metadata;
- IPFS-based off-chain storage for data retention that is scalable and impenetrable;
- TKMS for lifetime management and decentralized, granular key distribution.

III. METHODOLOGY

The proposed system uses TKMS, Brotli compression, AURA-CKKS homomorphic encryption, IPFS-based off-chain storage, and blockchain anchoring to safely, quickly, and privately handle car service records.

The implementation was carried out on an Intel Core i7-11700 processor with 16 GB RAM, running Windows 11. The AURA-CKKS framework was implemented in Python 3.10 using the SEAL/Pyfhel library. For the TKMS, Shamir Secret Sharing (SSS) was configured with $n = 5$ shares and threshold $t = 3$ using a 256-bit prime field. The blockchain environment used Ganache as the local Ethereum network, and IPFS web3storage for decentralized content storage. Figure 1 shows the flow of the proposed approach.

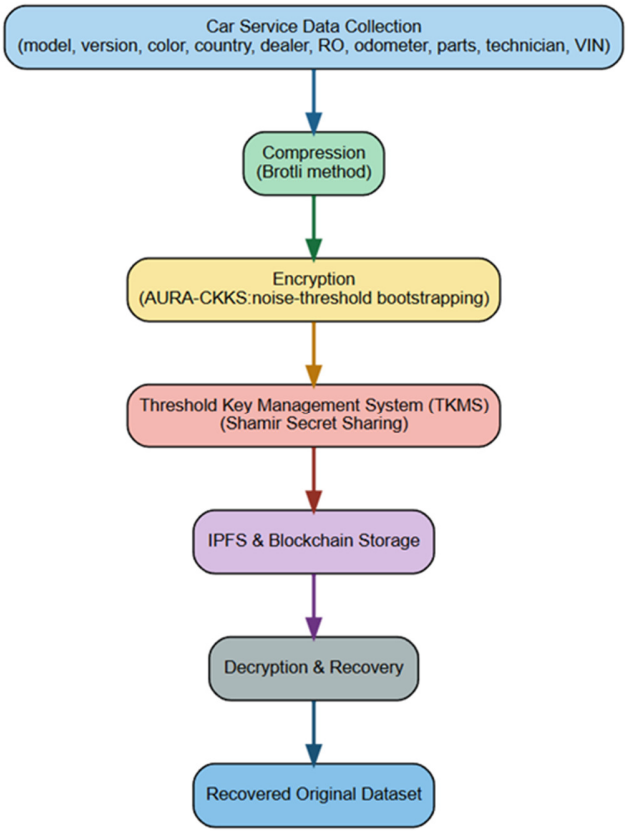

Fig. 1. Flow model of proposed work

A. Car Service Data Collection

The first step involves the systematic collection of vehicle service data from authorized dealers and warranty service providers. Each record contains structured attributes such as model type, version, color, country, dealer code, dealer, repair order date, odometer, part number, part name, technician description, and VIN corresponding to a specific vehicle. The dataset used in this study was obtained from FAXVIN [18] and Ford India Private Limited. Due to company confidentiality and proprietary restrictions, the dataset from Ford cannot be publicly shared. These attributes enable secure storage and traceability in a blockchain-based system. Formally, a single vehicle service record d_i is a tuple of attributes:

$$d_i = (mo_i, v_i, c_i, k_i, dc_i, de_i, ro_i, o_i, pn_i, pa_i, td_i, vin_i) \quad (1)$$

where mo_i denotes model type, v_i is the version, c_i is the color, k_i is the country , dc_i is the dealer code, de_i is the dealer entry, ro_i is the repair order, o_i is the odometer mark, pn_i is the part number, pa_i is the part name, td_i is the technician description, and vin_i in the VIN.

B. Compression

Before encryption, each data entry d_i was serialized and compressed using the lossless Brotli compression algorithm as

$$d_i^c = \mathrm{C}(d_i) \quad (2)$$

Brotli's adoption in blockchain-integrated systems has been demonstrated as an effective method for optimizing storage and gas usage without data loss [7]. It effectively reduces size by combining dictionary-based encoding, context modeling, and Huffman coding. The compression process converts the original record into its compressed form, where d_i^c is the compressed version of record i. The efficiency of compression is measured by Compression Ratio, defined as $CR = d_i/d_i^c$, with a CR greater than one indicating effective compression. This step reduces storage requirements and blockchain gas cost for storing encrypted records.

TABLE II. COMPRESSION OF DATA

Metric	Value
Original size	496 B
Compressed size	265 B
Compression ratio	0.57
Size reduction (%)	43.0%

As shown in Table II, a representative record of 496 B compresses to 285 B, achieving a 43.0% reduction. Despite the dataset's modest size, this compression step effectively reduces storage needs and blockchain gas costs during on-chain storage.

C. Encryption Phase

To ensure confidentiality and enable calculations on encrypted records, the compressed dataset is encrypted using the AURA-CKKS framework, which enhances CKKS with adaptive, noise-threshold-based bootstrapping. The plaintext d_i^c is encoded as a scaled polynomial $m_i$with a scaling factor Δ within the polynomial ring $R = \mathbb{Z}[x]/(x^N + 1)$ under ciphertext modulus q. Encryption with the public key pk generates the ciphertext pair $c_i = (c_{0i}, c_{1i})$, which combines the encoded message and randomized noise. AURA-CKKS monitors the noise and triggers bootstrapping only when a threshold is surpassed, reducing unnecessary computation on the blockchain-integrated system [19].

D. Threshold Key Management System (TKMS)

For key management, the proposed approach uses threshold secret sharing to provide robust and cooperative access control. The system does not store the secret key sk for AURA-CKKS encryption as a single entity after it has been generated. Instead, sk is split into n fragments using SSS, of which only a selection t (with $t \leq n$) is required for reconstruction. Specifically, the key is embedded as the constant term of a random polynomial of degree $t - 1$ as

$$f(x) = SK + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1} (mod\ q) \quad (3)$$

where a_j are randomly chosen coefficients. Each participant receives a share

$$S_i = f(i)\ ,\ \text{for } i = 1,2, \ldots n \quad (4)$$

During decryption, any t authorized participants can jointly reconstruct the secret key by Lagrange interpolation at $x = 0$.

$$sk = \mathrm{f}(0) = \sum_{i=1}^{t} S_i \prod_{\substack{1<j<t \\ j \neq t}} (0-j)/(i-j)\ \ (mod\ q) \quad (5)$$

Since no one has the entire key, this threshold design eliminates single points of failure and ensures collusion resistance up to $t-1$ compromised shares. Tamper-evident key usage is achieved by combining blockchain logging with threshold cryptography. Throughout its existence, the AURA-CKKS secret key (sk) is managed by the TKMS. If a share or participant is hacked, the compromised share is revoked on-chain, and a new sharing polynomial $f(x)$ is created using the same sk . Through relinearization, regular key renewal preserves ciphertext continuity. Lagrange interpolation reconstructs the sk at $x = 0$ without exposing it outside of secure memory, and valid shares are kept in a Hardware Security Module (HSM) for audit and recovery.

E. IPFS and Blockchain

To provide decentralized, tamper-resistant storage, the suggested system stores the encrypted dataset C_i in the InterPlanetary File System (IPFS). IPFS uses the file hash to create a Content Identifier (CID) for each record submitted. Since even a single bit change in the encrypted file results in a new CID, this CID guarantees immutability and integrity. The blockchain only keeps the metadata mapping, which consists of the following, rather than the entire encrypted record:

$$(h(VIN_i, CID_i, pk_i, \sigma_i,) \quad (6)$$

Here, $h(VIN_i) = H(VIN_i)$ is the hashed VIN, ensuring privacy while enabling unique mapping, CID_i is the IPFS content ID of the encrypted dataset, pk_i is the public key used for encrypting this dataset, and σ_i is the digital signature of the dealer or service provider, ensuring authenticity and non-repudiation. This design ensures that the VIN itself is never exposed on-chain, yet the system can still uniquely map records to their encrypted counterparts in IPFS.

F. Decryption

The encrypted-compressed dataset is safely recovered and rebuilt during the decryption procedure. The hash of the VIN $h(VIN_i)$ is used to query the blockchain for a particular entry, which acts as a unique index to locate the corresponding tuple $h(VIN_i, CID_i, pk_i, \sigma_i)$. The encrypted data C_i is retrieved from the IPFS using the stored CID_i. The compressed dataset is then obtained by applying the decryption process with the corresponding private key (or threshold rebuilt secret key), $d_i^c = Dec_{ski}(C_i)$. Subsequently, the compressed form d_i^c is decompressed using the Brotli algorithm to recover the original dataset:

$$d_i = Decompress\ (d_i^c) \quad (7)$$

Thus, the original record:

$$d_i = (mo_i, v_i, c_i, k_i, dc_i, de_i, ro_i, o_i, pn_i, pa_i, td_i, vin_i)$$

is reconstructed exactly as it was at the time of acquisition. This ensures that the data stored and retrieved via blockchain and IPFS maintains end-to-end confidentiality, integrity, and availability within the proposed framework.

G. Security Formalization and Threat Model

The suggested AURA-CKKS and TKMS framework's security is examined using semantic security Indistinguishability under a Chosen-Plaintext Attack (IND-CPA) paradigm, which was carried over from the original CKKS scheme. Since its adaptive scaling and bootstrapping processes only use ciphertexts and public evaluation keys, AURA-CKKS preserves the IND-CPA security that CKKS has demonstrated under the Ring Learning With Errors (RLWE) assumption, exposing no secret-dependent information [1]. Let the set of protocols used in the suggested encryption method be $AURA - CKKS = (KeyGen, Enc, Dec, Eval, Bootsrap)$. The following is how an adversary "A" engages with a challenger:

1. Key generation: To obtain the key pair, the challenger runs the key generation procedure. $pk, sk \leftarrow KeyGen()$. The public key pk is disclosed to the enemy "A", but the secret key sk is kept hidden.
2. Challenge messages: The adversary selects two plaintext messages of equal length, denoted m_0 and m_1.
3. Encryption challenge: the challenger selects a random bit $b \in (0,1)$, encrypts one of the messages to produce $c^* = Enc(pk, m_b)$, and sends c^* to the adversary.
4. Guess: The adversary outputs a guess bit b', indicating which plaintext was encrypted. The adversary advantage in distinguishing between the encryption of m_0 and m_1 is defined as $Adv_{AURA-CKKS}^{IND-CPA}(A) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Since ciphertexts cannot be discriminated more accurately than by random guessing, the method ensures semantic security by achieving IND-CPA security when the adversary's advantage is minimal for all Probabilistic Polynomial-Time (PPT) attackers. The system uses an honest-but-curious model, in which nodes adhere to the protocol but may attempt to deduce information from the data at hand. Table III summarizes the threat model for the AURA-CKKS with the TKMS framework.

The following section presents the unified AURA-CKKS with IPFS and blockchain-based secure storage and retrieval algorithm, followed by a concise notation, Table IV, that defines all symbols and variables used throughout the procedure.

TABLE III. THREAT MODEL AND DEFENCE FOR AURA-CKKS WITH TKMS

Adversary	Capability	Attack vector	Defence mechanism
Passive observer	Keeps an eye on cipher texts.	Statistical evaluation	IND-CPA security in AURA-CKKS under the RLWE
Malicious blockchain node	Replays or modifies on-chain data	Replay / substitution	Hash anchoring with blockchain immutability
Semi- honest evaluator	Performs homomorphic	Ciphertext pattern analysis	Randomization and relinearization
Key management adversary	Accesses key shares	Unauthoriz-ed key access	TKMS key controls and auditing

Algorithm 1: End-to-End Processing of Vehicle Record

Input: Vehicle record d_i

Output: Reconstructed record $\bar{d}_i$

1. Serialize d_i
2. Compute compressed record: $d_i^c = \mathrm{C}(d_i)$
3. Encode compressed plaintext $m_i = Encode(\ d_i^c, \Delta)$
4. Encrypt using AURA-CKKS: $c_i = Enc_{pk}(m_i)$
5. Compute noise budget: $B = Noice(c_i)$
6. If $B > \tau$ then $c_i \leftarrow Bootstrap(c_i)$ End If
7. Generate Shamir polynomial: $f(x) = SK + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1} (mod\ q)$
8. Compute shares: $S_i = f(i)\ , for\ i = 1,2, \ldots n$
9. Upload ciphertext to IPFS and obtain: $CID_i = IPFS.Add(c_i)$
10. Compute hashed identifier $h(VIN_i) = H(VIN_i)$
11. Generate dealer signature: $\sigma_i = Sign_{skd}(\mathrm{h}(VIN_i), CID_i)$
12. Write blockchain entry: $h(VIN_i, CID_i, pk_i, \sigma_i)$
13. Query blockchain using $h(VIN_i)$
14. Retrieve CID_i – obtain ciphertext from IPFS
15. Reconstruct secret key using any t shares: $sk = \sum_{i=1}^{t} S_i \prod_{\substack{1<j<t \\ j \neq t}} (0-j)/(i-j)\ (mod\ q)$
16. Decrypt ciphertext: $d_i^c = Dec_{ski}(C_i)$
17. Decompress: $d_i = Decompress\ (d_i^c)$

Return: $\bar{d}_i$

TABLE IV. NOTATION FOR THE PROPOSED ALGORITHM

Notation	Description
d_i	Input vehicle record
$\bar{d}_i$	Reconstructed output record
d_i^c	Compressed vehicle record
m_i	Encoded plaintext
Δ	Scaling factor
Enc_{pk}	Encryption using the public key
B	Noise budget
CID_i	IPFS content identifier for ciphertext
VIN_i	VIN
$\mathrm{H}(VIN_i)$	Hash function for VIN
$\mathrm{h}(VIN_i)$	Hashed identifier stored on-chain
$Sign_{skd}$	Digital signature using dealer's private key
σ_i	Dealer signature for the record
pk_i	Dealer or system public key stored on-chain
$Dec_{ski}(C_i)$	Decryption using reconstructed private key

IV. RESULTS

The framework's resistance to attacks, such as replay, data manipulation, Sybil, eavesdropping, and key theft, was assessed to determine its accuracy and efficiency. Figure 2 compares the resistance of three approaches, a baseline (on-chain storage), AURA-CKKS, and AURA-CKKS with TKMS, on a scale of 1 (weak) to 5 (strong). The baseline showed low resilience, particularly to eavesdropping and key theft (score=1). AURA-CKKS improved security, moderate to strong protection (scores in the 3-5 range), but remained slightly vulnerable to key theft (score=2). In contrast, AURA-CKKS+TKMS consistently scored high (4-5) across all attack types, continuously obtaining high resistance (scoring 4-5) in every attack category, offering strong defence against replay and key theft through decentralized key verification and adaptive bootstrapping. These results demonstrate the framework's ability to secure encrypted vehicle records and defend against common blockchain-related risks.

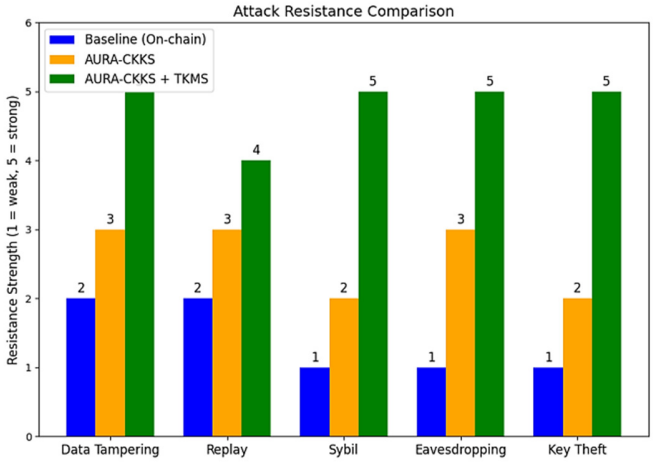

Fig. 2. Attack resistance comparison.

A. GAS Consumption

Figure 3 shows the gas consumption trends for storing an increasing number of VIN records on the blockchain using three distinct methods: (i) raw on-chain storage (baseline), (ii) AURA-CKKS-encrypted on-chain storage, and (iii) the suggested AURA-CKKS framework integrated with IPFS for off-chain storage.

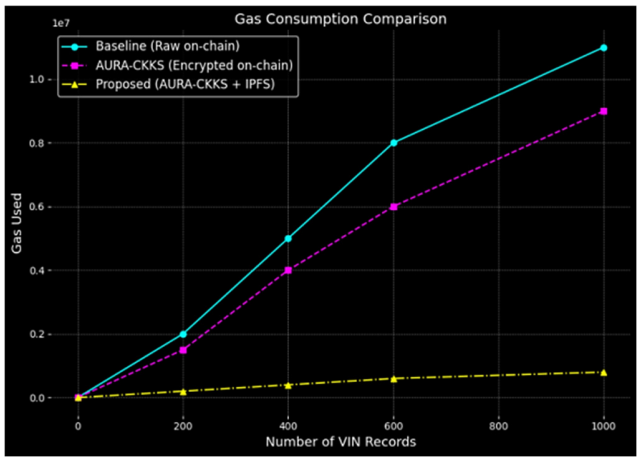

Fig. 3. Gas consumption comparison.

The baseline technique shows the steepest rise in gas usage, reaching almost 1.2×10^7 gas units for 1000 records. AURA-CKKS encryption reduces gas to about 9×10^6 gas units but remains relatively costly. In contrast, AURA-CKKS+IPFS drastically lowers gas usage to near zero for the same number of records by storing encrypted content off-chain and keeping only lightweight references on-chain. The blockchain records the hashed VIN, IPFS CID, public key, and dealers' signatures for verification, minimizing on-chain storage while ensuring secure verification, making it highly gas-efficient and scalable compared to baseline and AURA-CKKS approaches.

B. Execution Time and Analysis

Figure 4 compares the average confirmation time of the baseline on-chain approach, AURA-CKKS on-chain encryption, and AURA-CKKS+TKMS as the number of nodes on the blockchain increases. Each node (miner or validator) maintains a ledger copy, verifies transactions, and participates in consensus. Longer confirmation times arise from increased communication and consensus overhead. Processing a VIN record involves executing smart contract logic to store the hashed VIN, IPFS, CID, public key, and dealer's digital signature. The confirmation time measures the duration of the transaction, from its submission to validation by most nodes and inclusion in the block, making it immutable.

As shown in Figure 4, the AURA-CKKS modestly improves performance at about 10 s, while the baseline approach reaches 12 s at 100 nodes. The suggested AURA-CKKS+IPFS approach reduces execution and confirmation latency to 5.5 s at 100 nodes by minimizing on-chain payload and offering large encrypted data to IPFS, reducing the consensus and propagation cost of the network.

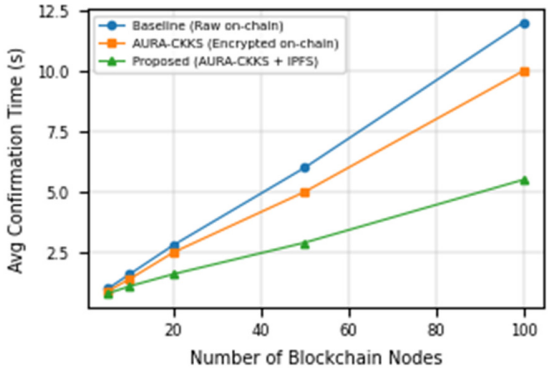

Fig. 4. Execution time vs number of nodes.

C. Transaction Load vs Confirmation Time

Figure 5 shows that the baseline raw on-chain approach has the highest confirmation time, reaching around 12 s for 1000 transactions. The AURA CKKS on-chain method reduces this to roughly 8 s, while the proposed AURA-CKKS+IPFS approach achieves the fastest performance at around 4 s. This improvement comes from reducing on-chain data size and parallelizable off-chain storage, which decreases the consensus and execution load on the blockchain network

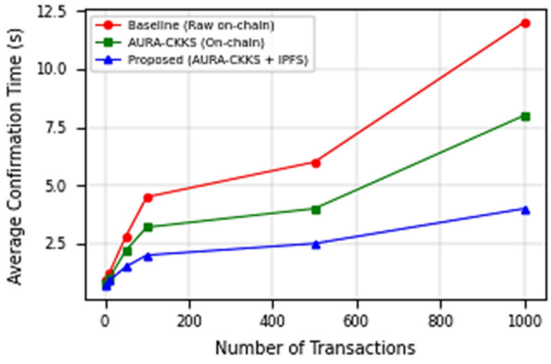

Fig. 5. Confirmation time vs transactions.

D. Query Load vs. Latency

A query refers to a read operation that retrieves a particular VIN record from the blockchain. Each query accesses the on-chain reference containing the hashed VIN, IPFS hash, and associated metadata, and if needed, fetches the encrypted data from IPFS for client-side decryption. Figure 6 illustrates how average latency varies with the number of concurrent requests for the three methods. Concurrent inquiries indicate that the blockchain network is processing several of these read requests at once, which puts a concurrent demand on the system and may lead to more resource contention. Latency is measured from the moment a query is issued until the system returns the response. Figure 6 reports the average latency per query. At 100 concurrent queries, the baseline approach shows a sharp increase, reaching about 8500 ms. The proposed AURA-CKKS+IPFS solution achieves the lowest latency at about 2000 ms due to its smaller on-chain data size and distributed off-chain retrieval, which reduce network congestion and accelerate responses under high query loads.

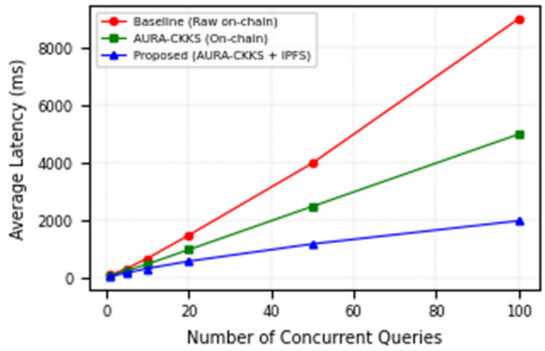

Fig. 6. Impact of concurrent queries on latency.

E. Limitations and Scope

Although the proposed TKMS offers threshold-based key management, its current specification is lacking in terms of operational control, auditability, and long-term key escrow. Recoverability over a period of ten to fifteen years is required for automotive data; however, the protocols for safely keeping, retrieving, and discarding escrowed keys, including administrative and legal constraints for multi-party access, are still not well defined. Moreover, there is no formalization of actual governance policies, such as who can manage the share lifetime, start key recovery, or authorize revocation. Lastly, the system lacks standardized gas-efficient methods for auditing revocation or renewal activities and confirming the validity of keys, although blockchain anchoring records important occurrences. For vehicle data management systems to remain safe, auditable, and compliant over time, these deficiencies must be addressed.

V. CONCLUSION AND FUTURE WORK

This study integrates blockchain, IPFS, and the AURA-CKKS homomorphic encryption technique to provide a safe and effective framework for storing and retrieving car service records. The proposed method maintains data confidentiality while drastically decreasing on-chain storage by just storing a small amount of metadata, such as the hashed car ID, IPFS CID, public key, and dealer's digital signature. To avoid unwanted access, sensitive data is encrypted and kept off-chain. By lowering the volume of on-chain data and transaction frequency, the framework also reduces gas usage. The experimental results prove that it has a compact blockchain footprint, low gas consumption, and consistent latency, even under large query loads, demonstrating a solid balance between security and performance. Furthermore, AURA-CKKS supports regulatory compliance by enabling privacy-preserving computation on encrypted data. The suggested system offers a decentralized car service record management solution that is scalable, economical, and privacy-focused.

Future research should focus on some important topics. First, more advanced access control and key management systems should be investigated to provide dynamic fine-grained data exchange among several stakeholders. Second, without jeopardizing data privacy, the incorporation of Zero-Knowledge Proofs (ZKPs) can offer further layers of verification. Third, AURA-CKKS's capability for intricate calculations on encrypted data at scale can be substantially enhanced by increasing its bootstrapping efficiency. A deeper understanding of the framework's potential for practical adoption can be obtained by implementing it on public blockchain networks, assessing its real-world performance and compatibility with current automobile industry standards. Future work also includes designing formal escrow protocols with multi-party access controls, developing on-chain revocation and audit primitives, and establishing governance playbooks to ensure secure, auditable, and long-term operation of vehicle data management systems. Lastly, to ensure long-term security against quantum attackers, future studies should investigate the integration of post-quantum cryptographic primitives.

REFERENCES

[1] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic Encryption for Arithmetic of Approximate Numbers," in *Advances in Cryptology – ASIACRYPT 2017*, vol. 10624, T. Takagi and T. Peyrin, Springer International Publishing, 2017, pp. 409–437.

[2] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "Bootstrapping for Approximate Homomorphic Encryption," in *Advances in Cryptology – EUROCRYPT 2018*, vol. 10820, J. B. Nielsen and V. Rijmen, Springer International Publishing, 2018, pp. 360–384.

[3] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic Evaluation of the AES Circuit," in *Advances in Cryptology – CRYPTO 2012*, vol. 7417, R. Safavi-Naini and R. Canetti, Springer Berlin Heidelberg, 2012, pp. 850–867.

[4] C. Gentry, S. Halevi, and N. P. Smart, "Better Bootstrapping in Fully Homomorphic Encryption," in *Public Key Cryptography – PKC 2012*, vol. 7293, M. Fischlin, J. Buchmann, and M. Manulis, Springer Berlin Heidelberg, 2012, pp. 1–16.

[5] T. V. T. Doan, M. L. Messai, G. Gavin, and J. Darmont, "A survey on implementations of homomorphic encryption schemes," *The Journal of Supercomputing*, vol. 79, no. 13, pp. 15098–15139, Sept. 2023, https://doi.org/10.1007/s11227-023-05233-z.

[6] C. Gouert, D. Mouris, and N. Tsoutsos, "SoK: New Insights into Fully Homomorphic Encryption Libraries via Standardized Benchmarks," *Proceedings on Privacy Enhancing Technologies*, vol. 2023, no. 3, pp. 154–172, July 2023, https://doi.org/10.56553/popets-2023-0075.

[7] F. Boemer, S. Kim, G. Seifu, F. D. M. de Souza, and V. Gopal, "Intel HEXL: Accelerating Homomorphic Encryption with Intel AVX512-IFMA52." arXiv, 2021, https://doi.org/10.48550/ARXIV.2103.16400.

[8] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System." arXiv, 2014, https://doi.org/10.48550/ARXIV.1407.3561.

[9] J. Alakuijala and Z. Szabadka, "Brotli Compressed Data Format," RFC7932, July 2016. https://doi.org/10.17487/RFC7932.

[10] Y. T. Jiang and H. M. Sun, "A Blockchain-Based Vehicle Condition Recording System for Second-Hand Vehicle Market," *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, Jan. 2021, Art. no. 6623251, https://doi.org/10.1155/2021/6623251.

[11] S. El-Switi and M. Qatawneh, "Application of Blockchain Technology in Used Vehicle Market: A Review," in *2021 International Conference on Information Technology (ICIT)*, Amman, Jordan, July 2021, pp. 49–54, https://doi.org/10.1109/ICIT52682.2021.9491670.

[12] J. Yuan, W. Liu, J. Shi, and Q. Li, "Approximate homomorphic encryption based privacy-preserving machine learning: a survey," *Artificial Intelligence Review*, vol. 58, no. 3, Jan. 2025, Art. no. 82, https://doi.org/10.1007/s10462-024-11076-8.

[13] D. Dhinakaran, L. Srinivasan, S. M. Udhaya Sankar, and D. Selvaraj, "Quantum-based privacy-preserving techniques for secure and trustworthy internet of medical things an extensive analysis," *Quantum Information & Computation*, vol. 24, no. 3 & 4, pp. 227–266, Mar. 2024, https://doi.org/10.26421/QIC24.3-4-3.

[14] D. Dhinakaran, N. J. Kumar, N. P. Ponnuviji, and B. P. Kumar, "Safeguarding confidentiality and privacy in cloud-enabled healthcare

systems with spectrasafe encryption and dynamic k-anonymity algorithm," *Expert Systems with Applications*, vol. 279, June 2025, Art. no. 127584, https://doi.org/10.1016/j.eswa.2025.127584.

[15] D. Dhinakaran, R. Ramani, S. E. Raja, and D. Selvaraj, "Enhancing clinical data security with the contextual polynomial-based data protection model (CPDPM)," *Biomedical Signal Processing and Control*, vol. 111, Jan. 2026, Art. no. 108329, https://doi.org/10.1016/j.bspc.2025.108329.

[16] D. Dhinakaran, L. Srinivasan, D. Selvaraj, and T. P. Anish, "A Novel Privacy Presevation of Healthcare Data with Information Entropy-Based Multi-Scheme Fully Homomorphic Encryption and Rivest Shamir Adleman Techniques," *Biomedical Engineering: Applications, Basis and Communications*, vol. 37, no. 04, Aug. 2025, Art. no. 2450060, https://doi.org/10.4015/S1016237224500601.

[17] D. Dhinakaran, G. Prabaharan, K. Valarmathi, S.M. U. Sankar, and R. Sugumar, "Safeguarding Privacy by utilizing SC-DℓDA Algorithm in Cloud-Enabled Multi Party Computation," *KSII Transactions on Internet and Information Systems*, vol. 19, no. 2, Feb. 2025, https://doi.org/10.3837/tiis.2025.02.014.

[18] "Vehicle History Reports," *FAXVIN*. https://www.faxvin.com/.

[19] S. Yasmin and G. S. Devi, "An Adaptive User-Guided Resilient Approach for Dynamic Bootstrapping in Homomorphic Encryption," *Engineering, Technology & Applied Science Research*, vol. 15, no. 4, pp. 25984–25992, Aug. 2025, https://doi.org/10.48084/etasr.12045.