

OrbitalChain: Truth Discovery Mechanisms for Dynamic Data Streams in Inter-Satellite Communication

Edson Eliezer da Silva Tavares, Qi Xia, Wei Zhang, Hu Xia, Jianbin Gao ^(✉), Bole Wilfried Tienin, Grace Mupoyi Ntuala, Mohammed Raouf Senouci, Toussaint Elvis Compaore

Abstract Real-time data streaming analysis plays an important role in modern applications. Several technologies have been used by researchers to improve data streaming. Among these technologies, satellite communication has been efficient for the task. The use of Low Earth Orbit (LEO) satellite networks for data streaming, particularly in remote and underserved regions has introduced challenges in ensuring data integrity, trust, and security for inter-satellite communication. Traditional centralized data verification methods are limited due to high latency, limited computational resources, and vulnerability to single points of failure. To address these limitations, this paper presents OrbitalChain, a decentralized data streaming system designed to manage data access control services without relying on intermediate trust entities. The proposed system aims to secure real-time data fusion in LEO satellite networks through a sharded blockchain architecture, thus facilitating multi-source information fusion. Furthermore, we introduce a blockchain-based truth discovery mechanism to prevent the dissemination of false information. This mechanism continuously evaluates the trustworthiness of satellite nodes, using historical data and reliability scores to identify and mitigate the impact of malicious or faulty nodes. The proposed mechanism implemented on shard blockchain, enables consensus on data validity across the distributed satellite network, ensuring only verified information reaches end-users. Additionally, OrbitalChain employs a D-Stream clustering-based truth discovery mechanism to improve data reliability. Moreover, an inter-shard PBFT (Practical Byzantine Fault Tolerance) consensus protocol ensures efficient cross-shard transactions while mitigating Byzantine faults. During the experiment, we evaluated OrbitalChain over an active blockchain network. Experimental results confirm that OrbitalChain effectively balances scalability, security, better transaction throughput, low latency, and real-time performance, making it an optimal solution for trust management in next-generation LEO satellite networks.

Keywords Blockchain; Inter-Satellite Network; Data Stream; Consensus Algorithm; Zero-Trust Security

1 Introduction

Satellite networks is reshaping data-driven applications in the current digital landscape due to the proliferation of data from Internet of Things (IoT) devices, particularly in time-critical environments that demand rapid and reliable data fusion [1]. The increasing complexity and volume of these data streams have underscored the need for robust and secure multi-source data management and fusion systems capable of supporting decision-making across diverse sectors, such as telecommunications, network security, and situational awareness in remote or inaccessible regions. LEO satellite constellations, with their extensive coverage and low-latency communication capabilities, are emerging as a pivotal infrastructure for aggregating, transmitting, and processing global data streams. Unlike terrestrial networks, which are constrained by physical geography and latency, LEO satellite networks offer resilient and high-speed data fusion. They enable efficient distribution capabilities that bridge the digital divide and provide coverage to underserved areas.

However, as satellite networks grow in scale and become essential to multi-source information fusion, ensuring the security, accuracy, and reliability of the data they transmit remains a significant challenge [2]. Traditional centralized data streaming architectures are vulnerable to single points of failure and cyber-attacks, posing risks to the integrity and availability of fused data streams. To address these limitations, this work presents OrbitalChain, a decentralized blockchain-based truth discovery framework that integrates LEO satellite

• Edson Eliezer da Silva Tavares, Qi Xia, Hu Xia, Jianbin Gao, Bole Wilfried Tienin, Grace Mupoyi Ntuala, Mohammed Raouf Senouci and Toussaint Elvis Compaore are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, 611731, China. E-mail: edsonskont@hotmail.com; xiaqi@uestc.edu.cn; xiahu@uestc.edu.cn; gaojb@uestc.edu.cn; tieninwilfried@std.uestc.edu.cn; gracemupoyi@gmail.com; senoucime-draouf@gmail.com; 202424080122@std.uestc.edu.cn.

• Hu Xia and Jianbin Gao are also with the State Key Laboratory of Blockchain and Data Security, Zhejiang University, HangZhou, 310027 ZheJiang, China. E-mail: xiahu@uestc.edu.cn; gaojb@uestc.edu.cn

• Wei Zhang is with the Key Laboratory of Intelligent Space TTC&O (Space Engineering University), Ministry of Education, Beijing, 100416, China. E-mail: zhangwei@hgd.edu.cn.

^(✉) Corresponding author: Jianbin Gao. E-mail: gaojb@uestc.edu.cn

Manuscript received: 2025-02-28; revised: 2026-01-23; accepted: 2026-05-18

networks to enhance the security and reliability of data streams in real-time applications. With this innovative convergence, we aim to create a more secure and efficient data streaming platform by leveraging the inherent strengths of blockchain's decentralized ledger and the expansive coverage of satellite constellations. Blockchain technology offers unparalleled advantages in data streaming by ensuring decentralization, security, and immutability [3]. Traditional data streaming platforms often rely on centralized architectures, which are susceptible to single points of failure and cyber-attacks. In contrast, a blockchain-based data streaming platform distributes data across a network of nodes, each maintaining a complete ledger of transactions. This decentralized approach not only mitigates the risk of data breaches but also enhances the resilience and fault tolerance of the system.

Blockchain's consensus mechanisms, such as Practical Byzantine Fault Tolerance (PBFT), ensure that all transactions are validated and recorded in an immutable ledger, providing a transparent and verifiable record of data streams. This transparency is crucial for applications requiring high levels of trust and security, such as financial transactions, healthcare data management, data streaming services and secure communications. Each transaction on a blockchain is cryptographically secured and validated by multiple nodes, making it virtually impossible to alter past records without consensus from the majority of the network. Moreover, blockchain's inherent transparency provides an additional layer of security, as all transaction logs and data streams can be audited and verified independently. This reduces the risk of data manipulation and enhances trust among network participants.

To further augment the security and reliability of data streams, we propose a novel method based on streaming reliability scores. Each satellite data source is evaluated and assigned a reliability score, which reflects its historical performance, trustworthiness, and the accuracy of the data it provides. These scores data are dynamically updated and maintained within the blockchain, ensuring that only data from reputable sources is streamed to end-users. By integrating these scores into the blockchain framework, we establish a zero-trust content paradigm within the system. Applying this satellite scoring system not only filters out unreliable or malicious data sources but also incentivizes satellites to maintain high standards of data integrity. This approach offers a more robust mechanism, ensuring data quality and trustworthiness in real-time

Thanks to advancements in onboard processing and storage technologies, modern satellites are increasingly capable of

serving as streaming servers. These satellites can process and store data locally, reducing the need to transmit large volumes of data back to ground stations for processing [4]. This capability allows them to act as decentralized data servers, reducing dependency on terrestrial data centers and minimizing latency. This capability not only reduces latency but also enhances the scalability of the data streaming platform. By acting as decentralized streaming servers, satellites can provide more resilient and efficient data services. This configuration allows for real-time data processing and streaming, improving the overall efficiency and responsiveness of the network.

Traditional centralized signature-based verification schemes rely on persistent connectivity to ground stations or central authorities. However, in LEO constellations, satellites are in constant motion, resulting in intermittent inter-satellite links and limited windows for communication with centralized entities. This renders centralized trust verification infeasible or delayed, compromising the system's responsiveness and fault tolerance in scenarios where timely trust assessment is critical. Static voting systems similarly fall short because they assume a fixed set of participants and predefined trust weights, which are ill-suited to dynamic satellite topologies where nodes may frequently join, leave, or malfunction due to environmental factors. Such static schemes do not adapt to the continuous data streaming and mobility-driven trust fluctuations inherent to inter-satellite networks. OrbitalChain addresses these limitations by implementing a decentralized, real-time trust and truth discovery mechanism, where each satellite peer contributes to the computation and validation of trust scores using secure multiparty computation and D-Stream clustering. This enables adaptive, autonomous decision-making that evolves with the network, ensuring that trust can be evaluated and maintained even in the face of orbital mobility, link disruptions, and peer churn.

The main contributions of this work can be summarized as follows:

- We introduce OrbitalChain, a decentralized blockchain-based data fusion framework that integrates LEO satellite networks to ensure secure, reliable multi-source data fusion in real-time applications.
- Our approach enables satellites to function as decentralized data servers equipped with advanced processing and storage capabilities. This reduces reliance on terrestrial data centers, minimizes latency, and enhances the scalability and efficiency of the data streaming platform.
- We propose a novel blockchain-based truth discovery mechanism to evaluate the trustworthiness of satellite

Table 1: Summary of existing works

Existing Works	Novelty	Shortcomings
Blockchain scalability via full sharding [5]	Block pipelining for concurrent transaction processing, improving throughput and consistency.	Relies heavily on efficient inter-shard communication, which can become a bottleneck with many shards; Assumes an honest majority within shards, which may not hold in adversarial environments.
Secure and scalable sharding [6]	Distributed randomness generation for fair shard assignment, ensuring security and unpredictability.	Introduces latency due to complex cryptographic operations; Overhead from randomness generation and shard reconfiguration, affecting overall efficiency.
High throughput blockchain [7]	Hybrid consensus mechanism combining PoW and PBFT, enhancing security and transaction efficiency.	Complex implementation requiring substantial computational resources; Energy inefficiencies due to reliance on PoW for initial shard formation.
Decentralized ledger with scale-out performance [8]	Atomic cross-shard transactions for secure inter-shard communication, microblocks for reduced latency.	Complexity in deployment and maintenance in large-scale networks; Overhead from frequent microblock generation and cross-shard coordination, affecting efficiency.
Distributed Authentication Protocol [9]	Integration of identity-based encryption (IBE) and blockchain for authentication in LEO networks. Elimination of certificate overhead in dynamic environments. Blockchain ensures data immutability and decentralization.	Latency issues are likely due to the protocol's reliance on complex verification processes, particularly in bandwidth-constrained and highly mobile environments.
IoT Data Privacy [10]	Use of blockchain and IPFS for IoT data privacy. Blockchain ensures tamper-resistant data storage, while IPFS enables decentralized content distribution. Enhanced data confidentiality and access control.	Scalability challenges are anticipated from the implementation of blockchain and IPFS, particularly concerning large IoT data volumes in distributed settings.
Chain Blockchain [11]	Development of private blockchains with federated consensus. Transaction verification and validation occur simultaneously. Implementation of sharding for scalability.	Limited accessibility to complete ledger state by every node. Potential scalability concerns arise from the implementation of sharding strategy.

nodes. Leveraging blockchain to dynamically assess and maintain trust levels, thereby enhancing the accuracy and integrity of the fused data.

- We developed a consensus method for the dynamic selection of the most reliable and honest satellite nodes to lead cluster sub-networks;

The rest of this paper is organized as follows: section 2 presents related works, section 3 introduces us to the existing preliminary concepts. Section 4 provides the details of our proposed satellite network based on sharding blockchain and its implementation. We show the system security analysis in section 5 and in section 6, we discuss our results and improvements. We finally conclude the work in section 8.

2 Related Works

Currently, limited research exists on the integration of blockchain technology with satellite systems, with few studies focusing on practical implementations or in-depth technical analysis. This gap presents a valuable opportunity for further exploration, as blockchain could address key challenges

in satellite networks, including secure data exchange and autonomous coordination of satellite constellations.

Sharding and scalability in blockchain technology have been extensively studied, with significant contributions enhancing both performance and security of distributed ledger systems. [5] introduced RapidChain, a protocol that significantly improves blockchain scalability through full sharding. Unlike traditional blockchain systems that suffer from scalability bottlenecks due to their reliance on a single chain for transaction processing, RapidChain divides the network into multiple smaller shards. Each shard processes transactions independently, thereby increasing the overall throughput. RapidChain employs a novel block pipelining mechanism that allows shards to process transactions concurrently while maintaining consistency across the network. However, RapidChain's performance heavily relies on efficient inter-shard communication, which can become a bottleneck as the number of shards increases. Additionally, the protocol assumes an honest majority within each shard, which may not hold in

highly adversarial settings.

[6] proposed a secure sharding protocol addressing both scalability and security challenges in open blockchains. Their protocol divides the blockchain network into multiple shards, each capable of processing its own subset of transactions. The protocol ensures security through cryptographic techniques and consensus mechanisms that prevent malicious actors from compromising the blockchain's integrity. One key contribution is the introduction of a distributed randomness generation protocol, ensuring fair and unpredictable shard assignment. However, the reliance on complex cryptographic operations can introduce latency, potentially impacting real-time processing capabilities. Furthermore, the overhead associated with randomness generation and shard reconfiguration can affect overall system efficiency.

[7] presented the Zilliqa project, aiming to achieve high throughput and low latency through network sharding. Zilliqa's approach involves dividing the network into shards, each capable of parallel transaction processing. This system leverages a hybrid consensus mechanism combining proof-of-work (PoW) with PBFT to enhance both security and efficiency. Zilliqa demonstrates potential to handle a high number of transactions per second (TPS), making it suitable for applications requiring rapid data processing. Nevertheless, the hybrid consensus mechanism can be complex to implement and may require substantial computational resources. Additionally, the reliance on PoW for initial shard formation introduces energy inefficiencies, limiting applicability in resource-constrained environments.

[8] proposed Omniledger, a decentralized ledger system designed to achieve scale-out performance through sharding. Omniledger employs a secure sharding mechanism ensuring data consistency and fault tolerance across the network. One key innovation is the use of atomic cross-shard transactions, allowing secure and efficient inter-shard communication. The system also introduces microblocks, reducing latency and improving transaction confirmation times. While Omniledger offers robust scalability and security features, its complexity poses challenges for deployment and maintenance in large-scale networks. Additionally, frequent microblock generation and cross-shard coordination can lead to increased overhead, potentially impacting the system's performance under high transaction loads.

[9] introduced a distributed authentication protocol tailored for LEO networks, combining identity-based encryption (IBE) and blockchain technology. By leveraging IBE, the protocol eliminates the need for certificates, thus reducing the overhead associated with key management in dynamic LEO

environments. However, the protocol's reliance on blockchain may introduce latency issues, especially in environments with limited bandwidth and high mobility, thus impacting real-time communication performance.

[10] presented an innovative approach to safeguarding IoT data privacy by leveraging blockchains and the InterPlanetary File System (IPFS). The integration of blockchain technology ensures transparent and tamper-resistant data storage, while IPFS facilitates decentralized and efficient content distribution. By utilizing blockchain for storing access control policies and cryptographic keys, the proposed scheme enhances data confidentiality and access control in IoT deployments. However, the scheme's reliance on blockchain and IPFS may introduce scalability challenges, particularly concerning the storage and retrieval of large volumes of IoT data in distributed environments.

Chain [11] specializes in developing private blockchains for storage purposes. It benefits from federated consensus. To add a block to the ledger, the majority of nodes also must vote for the blocks to be validated. The blocks' transactions are verified and validated simultaneously, which is different from other blockchains. Not every node has access to the complete ledger state. Then it employs a strategy that emphasizes sharding [12]. The Chain blockchain can expand by simultaneously validating transactions and enabling data copies based on sharding.

3 Preliminaries

3.1 Keplerian Orbits

Defining and maintaining the satellites orbits within a shard blockchain framework is critical for their function as streaming service nodes in our system. The orbital parameters must be precisely controlled to ensure stable and predictable trajectories, which in turn ensures reliable network performance, communication, coordination and efficient data handling. Keplerian orbits [13–16] are described by six orbital elements, which uniquely determine the size, shape, and orientation of the satellite's orbit in space. These elements are:

- (1) **Semi-major Axis (a):** The semi-major axis defines the size of the orbit. It is a half of the longest diameter of the elliptical orbit and is given by $a = \frac{r_a + r_p}{2}$. Where r_a is the apoapsis distance and r_p is the periapsis distance.
- (2) **Eccentricity (e):** it measures the deviation of the orbit from a perfect circle and is defined as $e = \frac{r_a - r_p}{r_a + r_p}$. The value of $e = 0$ corresponds to a circular orbit, whereas $e = 1$ denotes a parabolic trajectory.
- (3) **Inclination (i):** is the angle between the orbital plane and the equatorial plane of the Earth. It determines the

Table 2: Acronyms, variables, and parameters used in OrbitalChain

Symbol	Description	Symbol	Description
LEO	Low Earth Orbit	A	Number of antennas at each data stream consumer
LoS	Line of Sight	N	Satellite Antennas
NLoS	Non-Line of Sight	τ_c	Coherence time interval
DSC	Data Stream Consumer	τ_p	Uplink pilot transmission
DP	Data Provider	τ_u	Uplink data transmission
TLE	Two-Line Element	τ_d	Downlink data transmission
GPS	Global Positioning System	q_k	Pilot transmission power of a data stream consumer
AER	Azimuth-Elevation-Range	Ψ_k	Transmitted pilot signal matrix
UL	Uplink	$\sigma_{w^p}^2$	AWGN variance for pilot signals
DL	Downlink	$\sigma_{W_k^d}^2$	AWGN variance for downlink data
SINR	Signal to Interference plus Noise Ratio	$A_{m,k}$	MMSE estimation coefficient matrix
AWGN	Additive White Gaussian Noise	C_e	Channel estimation error covariance
MMSE	Minimum Mean Square Error	s_k	Data Symbol for a data stream consumer
TPS	Transactions Per Second	$V_{m,k}$	Precoding Matrix
PBFT	Practical Byzantine Fault Tolerance	$p_{m,k}$	Power scaling factor for transmission
SBFT	Smart Byzantine Fault Tolerance	x_m	Signal transmission from a specific shard
SLN	Satellite Leader Node	y_k	Received Signal
ANT	Satellite Active Node	W_k^d	AWGN vector at a data stream consumer
SANT	Satellite Semi-Active Node	$D_{k,i}$	Aggregate precoded channel from all shards
RT_x	Random Transaction Leader at time x	R_k	Spectral efficiency of k -th relay
MPC	Secure Multi-Party Computation	G_k	Expected precoded channel
SS	Secret Sharing	B_k	Interference Matrix
GC	Garbled Circuit	I	Identity matrix
MACs	Message Authentication Codes	l	Epoch
UAV	Unmanned Aerial Vehicle	x_l^k	Sensing value of a data provider at epoch l
D-Stream	Discretized Stream	S_i	A specific satellite
a	Semi-major Axis	Φ_l	Discovered truth at epoch l
e	Eccentricity	Φ_i	Share of truth held by satellite
i	Inclination	d_m	Squared-error distance function
Ω	Right Ascension	λ	Decay factor for historical data
ω	Argument of Periapsis	$st(k)$	Accumulated distance/state for a data provider
ν	True Anomaly	$[x]_i$	Secret share of value held by party i
r_a	Apoapsis Distance	$\mathbb{F}_q$	Finite field with large prime q
r_p	Periapsis Distance	$\mathbb{Z}_q$	Ring of integers modulo q
$\mathbf{h}$	Angular Momentum	n	Number of satellites/parties in protocol
$\mathbf{r}$	Satellite position vector	S	Set of corrupted party shares
$\mathbf{v}$	Satellite velocity vector	(a, b, c)	Beaver Triple
$\mathbf{n}$	Vector pointing to ascending node	Tri_0, Tri_1	Multiplication triplet sets
M	Total number of satellites in the network	x_j	secret share Data Provider
m	Index of specific satellite	$[t]_i$	Share of weighted sum
K	Total number of Data Stream Consumers	$[z]_i$	Share of weights
$\Omega_{m,k}$	Azimuth angle of a satellite	r	Random mask value for garbled circuit
$\theta_{m,k}$	Elevation angle of a satellite	C_{div}	Division Circuit
$d_{m,k}$	Distance between a satellite and a data stream consumer	S_0, S_1	Two-party computation satellites
$h_{m,k}$	Channel between k -th DSC and m -th satellite	π	Privacy-preserving streaming truth discovery protocol
$H_{m,k}$	Channel Matrix	PPT	Probabilistic Polynomial-time
$\hat{H}_{m,k}$	MMSE estimate of channel response	$\mathbf{x}$	Multi-dimensional data point
$L_{m,k}$	Path Loss	d	Number of dimensions in data space
$\kappa_{m,k}$	Ratio of LoS to NLoS	w	Data Weight
$\phi_{m,k}$	LoS phase	g	Density Grid
$\beta_{m,k}$	Large-scale LoS parameter	C_k	Set of data stream consumers sharing same pilots
$\lambda_{m,k}$	Large-scale NLoS parameter	x_j	secret share

tilt of the orbit and calculated as $i = \arccos\left(\frac{h_z}{|\mathbf{h}|}\right)$, where $\mathbf{h}$ is the specific angular momentum vector of the satellite and h_z is its z -component.

- (4) **Right Ascension of the Ascending Node (Ω):** is the angle from the reference direction to the ascending node (the point where the satellite crosses the equatorial plane from South to North). $\Omega = \arctan 2(n_y, n_x)$, where $\mathbf{n}$ is the node vector pointing towards the ascending node, with components n_x and n_y .
- (5) **Argument of Periapsis (ω):** is the angle from the ascending node to the periapsis, measured in the direction of the satellite's motion. $\omega = \arctan 2\left(\frac{\mathbf{e} \cdot \mathbf{k}}{|\mathbf{e}|}, \frac{\mathbf{n} \cdot \mathbf{e}}{|\mathbf{n}||\mathbf{e}|}\right)$, where $\mathbf{e}$ is the eccentricity vector and $\mathbf{k}$ is the unit vector along the z -axis.
- (6) **True Anomaly (ν):** is the angle between the periapsis and the satellite's current position, measured at the focus of the ellipse (the center of the Earth). $\nu = \arctan 2\left(\frac{\mathbf{r} \cdot \mathbf{v}}{|\mathbf{r}||\mathbf{v}|}, \frac{\mathbf{e} \cdot \mathbf{r}}{|\mathbf{e}||\mathbf{r}|}\right)$, where $\mathbf{r}$ is the position vector of the satellite, and $\mathbf{v}$ is its velocity vector.

3.2 Truth Discovery Over Data Streams

Truth discovery in streaming data, often termed as streaming truth discovery, involves dynamically aggregating sensory data from crowdsensing applications while respecting historical aggregation results [17]. Unlike traditional truth discovery algorithms that process static datasets in batches, streaming truth discovery algorithms continuously scan incoming sensory data at each epoch to incrementally update the veracity of crowdsensing tasks and the weights assigned to clients, integrating historical results judiciously. Our data streaming truth discovery is based on the recent streaming truth discovery framework proposed by Li et al. [17]. Initially, all client weights and accumulated distances are initialized to 1 and 0 respectively. During each epoch $l \geq 1$, the algorithm updates each client's weight w_k and calculates the truth Φ_l using a weighted aggregation approach. Subsequently, the accumulated distances of all clients are updated, where the distance function $d_m = (x_l^k - \Phi_l)^2$ typically adopts the squared-error metric to measure discrepancies. The parameter λ serves as a decay factor, emphasizing recent data's significance in weight estimation, aligning with the principles of weighted moving averages. After each epoch, client weights w_k are recalculated, ensuring that the streaming truth discovery process adapts dynamically to evolving data streams while preserving the integrity of historical aggregation outcomes.

4 Proposed Solution and Implantation

4.1 System Architecture

Our Proposed system in figure 1 leverages LEO satellites, storage servers, ground stations, and data users, creating a robust and efficient data transmission network.

LEO satellites function as streaming servers within the network, facilitating the transmission of data streams from storage servers to clients. These satellites operate as nodes within a shard blockchain network. The network is divided into multiple shards, each consisting of a cluster of interconnected satellites designated as $\{\text{Shard-1}, \dots, \text{Shard-}n\}$. As we see in figure 2, each shard is responsible for handling data streams for a specific geographic area or a specific portion of the network, allowing for parallel processing and increased throughput. Satellites in different shards may have Line-of-Sight (LoS) to specific geographic areas, allowing clients in those areas to connect to the satellite shard network corresponding to their location. Ensuring unobstructed signal paths between satellites, ground stations, and user terminals.

Storage servers, acting as Data Providers (DP), send data streams to the satellites for distribution across the network. These servers act as the source of data streamed to clients via the satellites. Data centers or random user clients can assume the role of DP.

Clients equipped with user terminals, can receive data streams directly from satellites, bypassing the need for ground stations, enhancing data transmission efficiency and reducing latency. These terminals are equipped with advanced antennas for direct communication with satellites. Ground stations serve as intermediary hubs, uplinking data streams from DP to satellites, and relaying data to clients who do not have direct satellite communication capabilities. OrbitalChain solution can also support other wide range of dynamic data stream use cases, such as maritime and UAV networks.

Blockchain network plays a critical role in our system, ensuring the integrity and efficiency of data transmission. By storing each satellite's truth discovery score on the blockchain, the system can prioritize honest satellite nodes for serving as streaming servers and shard leaders. This prioritization mechanism enhances the reliability and trustworthiness of the network, as satellites with higher truth discovery scores are more likely to be selected for critical roles. The peer-to-peer communication feature of blockchain is leveraged to improve network communication among satellites. The communication protocol ensures robust and efficient data exchange, eliminating single points of failure and enhancing the network's resilience. Each satellite operates as a node

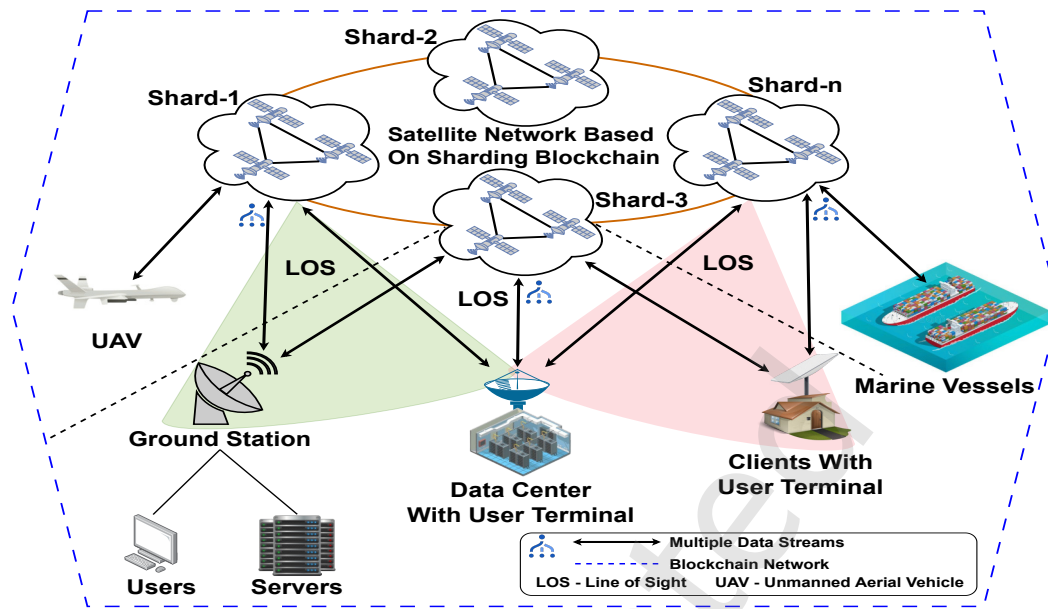


Fig. 1: OrbitalChain system architecture

within a shard blockchain network, which is geographically divided into multiple shards. Each shard is responsible for managing data streams for a specific region, enabling scalable and parallel processing across the network. The shards operate independently but are interconnected through a consensus mechanism, ensuring that the entire network remains synchronized and cohesive. Satellites with higher scores, reflecting their consistent performance and trustworthiness, are given preference in the leader selection process. This method ensures that the most reliable nodes are chosen for leadership roles. Also, another condition in our system is that only the satellites active at the precise moment of data streaming will participate in the streaming service. This dynamic participation ensures efficient resource utilization and adapts to the real-time availability of satellites.

To resolve the blockchain storage issue, we designed our system so that only the satellite's truth discovery scores and transaction logs are stored on the blockchain. The data streams are processed using D-Stream, which can handle perfectly the large volumes of real-time data. With this method, we keep the blockchain lightweight and efficient, ensuring transparency and accountability without storing entire data streams. By leveraging D-Stream for data processing, the network maintains high performance and scalability, while the blockchain ensures secure and immutable records of satellite performance and transaction history.

Our system data flow begins with data submission from DPs to satellites acting as streaming servers. Each data sensing value is split into secret shares using additive secret sharing

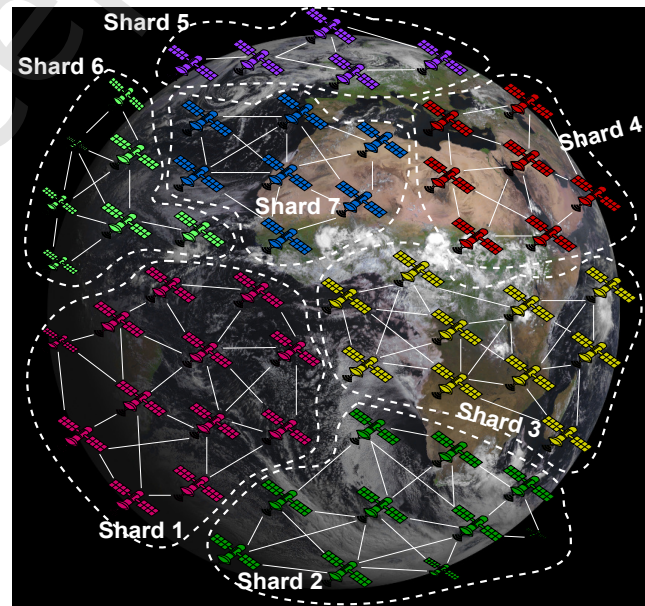


Fig. 2: Shard geographic areas

and securely transmitted to satellites in the same shard. Simultaneously, multiplication triplets (Beaver triples) and garbled circuit parameters are generated and distributed to support downstream secure arithmetic operations, particularly for nonlinear functions like division and logarithm. Once the shares are received, the MPC layer initiates a privacy-preserving streaming truth discovery. Following the secure truth computation, each satellite transmits and updated the share weights. The final phase of integration occurs at the blockchain layer, where truth values, trust scores are recorded

as transactions. Satellites synchronize their results using intra and inter-shard PBFT consensus, which guarantees Byzantine-resilient finality before committing state updates.

In order to establish a robust zero-trust environment within OrbitalChain, all components within the ecosystem are required to be registered and participate in the blockchain network. This registration ensures that every entity is authenticated and authorized to engage in data transactions. Integrating zero-trust with blockchain's immutable ledger creates a synergistic effect, enhancing both security and transparency. Each interaction and data transaction is recorded on the blockchain, providing a verifiable audit trail that supports compliance and forensic investigations.

4.2 Security Model

Our approach employs a threshold secret sharing scheme, combined with secure multi-party computation (MPC) techniques: Beaver triples and garbled circuits, to protect sensitive data during computation and data sharing. We define the security model in terms of the following entities:

- Satellites are semi-honest adversaries. Let n be the number of satellites involved in the protocol. Each holding secret-shared values $[x_1]_i, [x_2]_i, \dots, [x_m]_i$ for $i = 1, 2, \dots, n$. Satellites follow the protocol but they may try to infer information from the protocol messages.
- A share $[x]_i$ reveals no information about x , assuming the adversary has access to one share but not all shares.
- An adversary may corrupt up to t parties (where $t < n$) and gain full access to their shares. The adversary's goal is to infer information about the original secret values $x_1, x_2, \dots, x_m$.
- Beaver's multiplication triples $(a, b, c) \in F_q^3$ where $a \cdot b = c$, are used to ensure the security of multiplications of shared values.
- Garbled circuits are used for secure division and logarithm operations, ensuring that intermediate results are not leaked.

4.2.1 Formal Security Definitions

The following definitions formalize the security notions underlying our privacy-preserving streaming truth discovery protocol analysis (Section 5.2).

Definition 4.1. A semi-honest (passive) adversary $\mathcal{A}$ controls a subset of corrupted parties $\mathcal{C} \subset \{S_0, S_1, \dots, S_{n-1}\}$ where: (i) corrupted satellites execute the protocol faithfully according to specification, (ii) $\mathcal{A}$ observes all internal states, randomness, and messages of corrupted parties, and (iii) the corruption set $\mathcal{C}$ is fixed before protocol execution begins.

Definition 4.2. Let π denote the two-party protocol specified in Algorithm 2. The view of satellite S_i during protocol execution on inputs $\vec{x} = (x_l^1, \dots, x_l^K)$ at epoch l is defined as:

$$\text{View}_i^\pi(\vec{x}) = (x_i, r_i, \{m_j^i\}_{j \in [L]}) \quad (1)$$

where x_i represents S_i 's input (received shares), r_i is S_i 's internal randomness, and m_j^i represents the j -th message received by S_i during the protocol. For a corrupted set $\mathcal{C}$, the joint view is $\text{View}_{\mathcal{C}}^\pi(\vec{x}) = \{\text{View}_i^\pi(\vec{x})\}_{i \in \mathcal{C}}$.

Definition 4.3. Protocol π securely realizes an ideal functionality $\mathcal{F}$ if for every probabilistic polynomial-time (PPT) semi-honest adversary $\mathcal{A}$ corrupting parties in $\mathcal{C}$, there exists a PPT simulator $\mathcal{S}$ such that:

$$\{\text{IDEAL}_{\mathcal{F}, \mathcal{S}}(1^\kappa, \vec{x})\}_{\kappa, \vec{x}} \stackrel{c}{=} \{\text{REAL}_{\pi, \mathcal{A}}(1^\kappa, \vec{x})\}_{\kappa, \vec{x}} \quad (2)$$

where $\stackrel{c}{=}$ denotes computational indistinguishability, κ is the security parameter, $\text{IDEAL}_{\mathcal{F}, \mathcal{S}}$ denotes the output distribution in the ideal world where a trusted party computes $\mathcal{F}$, and $\text{REAL}_{\pi, \mathcal{A}}$ denotes the output distribution in the real protocol execution.

4.2.2 Threat Model

The system assumes that while most nodes behave correctly, a subset may be compromised, misbehave, or attempt to learn or manipulate private data. To ensure the privacy, robustness, and integrity of computations and consensus in this semi-trusted environment, OrbitalChain explicitly considers three adversarial types: passive, active, and Byzantine adversaries.

- **Passive Adversaries (Semi-Honest/Honest-but-Curious):** include compromised satellites with curious operators, data providers seeking to infer other participants' data, and ground station operators performing traffic analysis. The OrbitalChain framework mitigates this threat through threshold secret sharing, Beaver triples, and garbled circuits, which preserve input confidentiality and prevent information leakage during secure multiparty computations.
- **Active Adversaries (Malicious but Detectable):** include malicious data providers injecting/manipulating false data, compromised satellite operators causing protocol deviations, and man-in-the-middle attackers manipulating inter-satellite communications. OrbitalChain defends against such behavior using PBFT consensus to tolerate misbehavior, truth discovery scoring with D-Stream clustering to reduce the impact of untrustworthy nodes, and blockchain-based reputation tracking to penalize dishonest participants.

• **Byzantine Adversaries (Arbitrary and Colluding):**

These include nation-state actors with sophisticated satellite attack capabilities, compromised satellite manufacturers introducing systematic vulnerabilities, hostile satellite operators controlling multiple satellites, and coordinated multi-satellite attack coalitions. These adversaries behave arbitrarily, possibly colluding to disrupt consensus or computations. OrbitalChain assumes up to $n/3$ Byzantine satellites per shard and maintains correctness using PBFT consensus, enforces consistency via blockchain audit trails, and secures MPC through threshold cryptography with geographic satellite diversity to prevent collusion-based breaches.

Let's consider that an adversary obtains access to t shares, i.e., a subset $S \subseteq \{[x]_1, [x]_2, \dots, [x]_n\}$, where $|S| = t$. The adversary's goal is to infer the value of the secret x using these t shares. The privacy guarantee of the secret-sharing scheme asserts that the adversary gains no information about x , even with knowledge of t shares. The security of our scheme relies on the following assumptions:

Assumption 4.1. Independence of Shares: The shares are assumed to be independently generated and uniformly random in Z_q . This is crucial for ensuring that partial information about the original values cannot be inferred from the shares.

Assumption 4.2. Large Prime Modulus q : The prime modulus q is chosen to be sufficiently large to prevent modular wraparound issues and to ensure that the shares are uniformly distributed over Z_q .

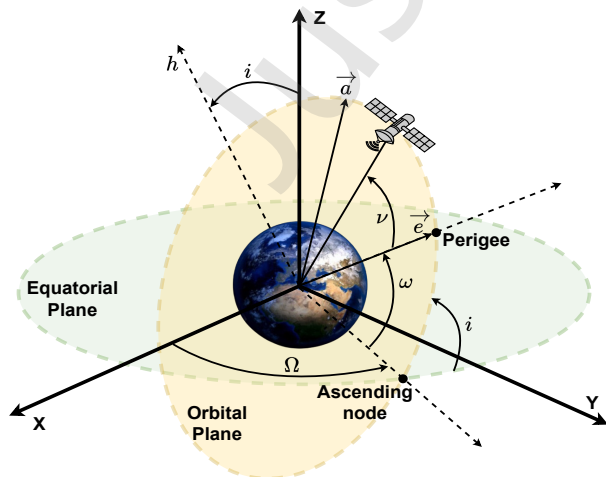


Fig. 3: Satellite's orbit

4.3 Network Model

We consider the downlink (DL) operations of a Low Earth Orbit (LEO) satellite network consisting of M satellites, indexed by $M = 1, 2, \dots, m$, visible to a set of k Data Stream Consumers (DSCs), indexed by $K = 1, 2, \dots, k$.

As depicted in Figure 3, the satellites' orbits are defined using Keplerian elements, which comprise six parameters (Section 3.1). These elements are derived from Two-Line Element (TLE) files publicly available for satellites. Accurate control of these Keplerian elements is crucial for ensuring the effective operation of the satellite cluster as shard blockchain nodes, facilitating synchronization and stable formation flying, which are vital for the reliability of the streaming service nodes. Continuous monitoring and adjustment of each satellite's position and velocity vectors are essential to maintain desired orbital parameters, optimizing the relative distances between satellites for efficient communication and data relay within the shard blockchain network. Each satellite synchronizes with the shard in its current position. When a satellite moves to another shard's area, it will connect with the shard corresponding to its new location. To determine the satellite's position relative to a DSC, we employ the azimuth-elevation-range (AER) coordinate system. Specifically, the position of the m -th satellite relative to the k -th DSC is described by the tuple $(\Omega_{m,k}, \theta_{m,k}, d_{m,k})$, representing azimuth, elevation, and range, respectively. For localizing the DSCs, we utilize a geodetic coordinate system with the World Geodetic System 1984 (WGS 84) [18, 19] reference ellipsoid, the standard for the Global Positioning System (GPS). The position of the k -th DSC is specified by the tuple $(\text{Latitude}_k, \text{Longitude}_k, \text{Height}_k)$, where Latitude_k is within the range $([-90^\circ, 90^\circ])$, Longitude_k within the range $[-180^\circ, 180^\circ]$, and Height_k represents height above the reference ellipsoid in meters.

4.4 Network Channel

We posit that the channel conditions remain constant within a coherence time interval encompassing τ_c samples. This assumption implies the adoption of a flat fading model for the small-scale fading, effectively neglecting frequency selectivity. The communication links between the satellites and the DSC comprise both LoS and non-line-of-sight (NLoS) components. The likelihood of encountering a LoS path increases proportionally with the elevation angle, reaching its maximum at an elevation of 90° as indicated in [20]. Consequently, we represent the channel between the k -th DSC and the m -th satellite using a Rician fading model, as detailed in [21]. The specific channel characterization can be computed as follows:

$$h_{m,k} = \sqrt{L_{m,k}} \left(\sqrt{\frac{\kappa_{m,k}}{\kappa_{m,k} + 1}} e^{j\phi_{m,k}} + \sqrt{\frac{1}{\kappa_{m,k} + 1}} h'_{m,k} \right) \quad (3)$$

In this equation 3, The Rician K -factor, denoted by $\kappa_{m,k}$, quantifies the ratio of LoS to NLoS power. We assume the NLoS component exhibits Rayleigh fading, with the corresponding channel coefficient $h'_{m,k}$ following a complex Gaussian distribution with zero mean and unit variance, i.e., $h'_{m,k} \sim \mathcal{CN}(0, 1)$. The LoS component is characterized by a phase term $\phi_{m,k}$, uniformly distributed within the interval $[-\pi, \pi]$, i.e., $\phi_{m,k} \sim \mathcal{SU}[-\pi, \pi]$. Considering the relative movement between the satellites and DSCs, we assume that the Doppler shift is effectively mitigated through a synchronization process.

For ease of notation, we rewrite Equation (3) as:

$$h_{m,k} = \sqrt{\beta_{m,k}} e^{j\phi_{m,k}} + \sqrt{\lambda_{m,k}} h'_{m,k}, \quad (4)$$

where

$$\beta_{m,k} = \frac{\kappa_{m,k}}{\kappa_{m,k} + 1} L_{m,k}, \quad \lambda_{m,k} = L_{m,k} / (\kappa_{m,k} + 1). \quad (5)$$

Therefore, the large-scale parameters $\beta_{m,k}$ and $\lambda_{m,k}$ in eq. (4) change slowly and can be known a priori.

4.5 Shard Network Connectivity

In our connectivity model, satellites are organized into clusters, termed shards, each overseen by a shard leader. These leaders facilitate and validate cooperative data transmission (Section 4.8) among the shards to DSCs. This arrangement effectively implements a Multiple Data Stream system, as detailed in [22] and [23] (Fig. 1).

We posit that the underlying transmission protocol employs time division duplexing (TDD). Consequently, the coherence time interval is partitioned into three sub-intervals: τ_p, τ_u, τ_d . The initial τ_p samples are allocated for uplink (UL) pilot transmission. The following τ_u samples are dedicated to UL data transmission, while the remaining τ_d samples are reserved for downlink (DL) data transmission. The UL channel is estimated based on the UL pilots, and this estimate is subsequently employed for precoding DL data transmissions. This is feasible due to the principle of channel reciprocity, which ensures the validity of the UL channel estimate for the DL direction as well.

We assume that the DSCs and satellites in shards are equipped with A and N antennas, respectively. Therefore,

the channel response matrix between the k -th DSC and m -th shard is $H_{m,k} \in \mathbb{C}^{A \times N}$, and is given by:

$$H_{m,k} = \sqrt{\beta_{m,k}} e^{j\phi_{m,k}} + \sqrt{\lambda_{m,k}} H'_{m,k}, \quad (6)$$

Note that, the channel matrix denoted as $H_{m,k}$, is a complex-valued matrix of dimension $A \times N$. Its elements are independent and identically distributed (i.i.d.) circularly symmetric complex Gaussian random variables (RVs) with zero mean and unity variance. In this scenario, we assume that the LoS components of the received signals at different antennas are identical due to the significant distance between the satellite and the DSC relative to the spacing between the antennas.

4.5.1 Channel Setup

We estimate the UL channel at the shards using the UL pilots. We assume that the pilots employed for the N antennas of each DSC are orthogonal. Nevertheless, this set of pilots can be reused for other DSCs, leading to pilot contamination. We designate the subset of DSCs that utilize the same set of pilots as DSC _{k} as $\mathcal{C}_k$. The transmitted pilot signal from DSC _{k} is represented by $\sqrt{q_k} \Psi_k \in \mathbb{C}^{\tau_p \times N}$, where q_k denotes the pilot power of DSC _{k} and $\Psi_k^H \Psi_k = \tau_p I_N$, with I_N being the identity matrix of size N .

The signal received at the m -th Shard from all DSCs can be computed as follows:

$$Y_m^P = \sum_{k=1}^K \sqrt{q_k} H_{m,k} \Psi_k^H + W_m^P, \quad (7)$$

We denote the received pilot signals at the A antennas of the m -th Shard by $Y_m^P \in \mathbb{C}^{A \times \tau_p}$. Additive white Gaussian noise (AWGN) is modeled as a matrix $W_m^P \in \mathbb{C}^{A \times \tau_p}$, where its elements are independent and identically distributed (i.i.d.) complex Gaussian random variables (RVs) with zero mean and variance $\sigma_{w^p}^2$. To estimate the UL channel of the k -th DSC, we derive sufficient statistics from the received signal.

$$\begin{aligned} Y_{m,k}^P &= Y_m^P \Psi_k = \sum_{i=1}^K \sqrt{q_i} H_{m,i} \Psi_k^H \Psi_i + W_m^P \Psi_k \\ &= \sqrt{q_k} \tau_p H_{m,k} + \sum_{k' \in \mathcal{C}_k \setminus \{k\}} \sqrt{q_{k'}} \tau_p H_{m,k'} + W_{m,k}^P. \end{aligned} \quad (8)$$

This can be derived given that

$$\Psi_k^H \Psi_{k'} = \begin{cases} \tau_p I_N, & k' \in \mathcal{C}_k \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

Pilot contamination arises when DSCs utilize the same pilot sequence as DSC_k. This contamination enables the estimation of the UL channel using techniques like minimum mean square error (MMSE) and least squares estimators [24], or by employing machine learning algorithms as demonstrated in [25].

For this study, we adopt a phase-aware MMSE estimation approach. The derivation of the MMSE estimate of the channel response is presented in Lemma 4.1.

Lemma 4.1. *The phase-aware MMSE estimate of the channel response, $H_{m,k}$, is given as:*

$$\hat{H}_{m,k} = \sqrt{\beta_{m,k}} e^{j\phi_{m,k}} + \lambda_{m,k} \sqrt{q_k} \tau_p \left(Y_{m,k}^P - \sum_{i \in C_k} \sqrt{q_i} \tau_p \sqrt{\beta_{m,i}} e^{j\phi_{m,i}} \right) A_{m,k}, \quad (10)$$

where

$$A_{m,k} = \left(\sum_{i \in C_k} q_i \tau_p^2 \lambda_{m,i} I_N + \sigma_{w^p}^2 I_N \right)^{-1}. \quad (11)$$

The error covariance matrix is given by:

$$C_e = \mathbb{E} \left\{ \left(H_{m,k} - \hat{H}_{m,k} \right)^H \left(H_{m,k} - \hat{H}_{m,k} \right) \right\} \quad (12)$$

$$= A \lambda_{m,k} - A \lambda_{m,k}^2 q_k \tau_p^2 A_{m,k}.$$

4.5.2 Downlink Setup

Data is transmitted from the Shards to the DSCs and clients in the DL direction. Each Shard transmits the same symbol to all data requesters. The symbol intended for the k -th DSC is denoted by $s_k \in \mathbb{C}^{N \times 1}$, where each element is a complex Gaussian random variable with zero mean and unit variance. Consequently, the autocorrelation of the symbol is given by $\mathbb{E} s_m s_m^H = I_N$. Prior to transmission, each symbol is precoded using the matrix $V_{m,k} \in \mathbb{C}^{A \times N}$. For example, in the case of conjugate beamforming (CB), the precoding matrix of the m -th Shard for the k -th DSC is:

$$V_{m,k} = \sqrt{p_{m,k}} \hat{H}_{m,k}, \quad (13)$$

The power scaling factor is denoted by $p_{m,k}$ and $\hat{H}_{m,k}$ represents the estimated UL/DL channel. The transmitted signal from the m -th Shard to all DSCs, denoted by $x_m \in \mathbb{C}^{A \times 1}$, is defined as follows:

$$x_m = \sum_{k=1}^K V_{m,k} s_k. \quad (14)$$

Accordingly, the received signal at the k -th DSC is calculated by

$$y_k = \sum_{m=1}^M H_{m,k}^H x_m + W_k^d = \sum_{i=1}^K D_{k,i} s_i + W_k^d, \quad (15)$$

We denote the AWGN vector at the k -th DSC by $W_k^d \in \mathbb{C}^{N \times 1}$. The precoded channel from all Shards to the k -th DSC is represented by $D_{k,i} = \sum_{m=1}^M H_{m,k}^H V_{m,i}$. As the DSC lacks prior knowledge of the precoded channel gain, it is approximated using either the mean value (statistical channel state information), $\mathbb{E} D_{k,k}$, or an estimate. We assume that the DSC utilizes the mean value, consistent with several studies in the literature (e.g., [26], [27]). Consequently, the received signal can be expressed as follows:

$$y_k = \mathbb{E}\{D_{k,k}\} s_k + (D_{k,k} - \mathbb{E}\{D_{k,k}\}) s_k + \sum_{i=1, i \neq k}^K D_{k,i} s_i + W_k^d, \quad (16)$$

where, the components in question represent the desired, unknown, interference, and noise signals, respectively. Consequently, the spectral efficiency of the k -th Decode-and-Forward (DF) relay can be expressed as:

$$R_k = \frac{\tau_c - \tau_p}{\tau_c} \log_2 (|I_N + \text{SINR}_k|), \quad (17)$$

where interference ratio (SINR) is

$$\text{SINR}_k = G_k^H B_k^{-1} G_k, \quad (18)$$

$$G_k = \mathbb{E}\{D_{k,k}\} = \mathbb{E} \left\{ \sum_{m=1}^M H_{m,k}^H V_{m,k} \right\}, \quad (19)$$

$$B_k = \mathbb{E} \left\{ \sum_{m=1}^M \sum_{n=1}^M \sum_{i=1}^K H_{m,k}^H V_{m,i} V_{n,i}^H H_{n,k} \right\} - G_k G_k^H + \sigma_{W_k^d}^2 I_N. \quad (20)$$

4.6 Data Clustering

In our system, we use D-Stream (Discretized-Stream) algorithm (Alg. 1) to deal with the clustering data stream protocols. By aggregating multiple data sources as one, D-Stream [28] can efficiently perform the clustering process. D-Stream algorithm consolidates incoming data from various sources into a single entity at the same time, aggregating

them and starting a clustering procedure [29]. Each unit of data holds information which represents the characteristics of an object at a particular time stamp. Once D-stream detects a data injection, the algorithm captures each incoming data point $x = (x_1, x_2, \dots, x_d)$ along with its associated weight w . Each data point represents an object in a multi-dimensional space, and its weight indicates its significance in the clustering process. The clustering technique is similar to the standard approach used for density-based clustering, once the density of the grids is calculated at the given time. The algorithm produces the initial cluster after it reaches a predefined time distance. Then, the algorithm eliminates intermittent grids regularly, in case there is no data injection between the current time gap and adjusts the clusters. Our clustering data streams algorithm also updates the characteristic vector of grid g by incrementing its density with the weight w of the data point. Additionally, it updates the sum of coordinates and the sum of squares of the coordinates for each dimension i , weighted by w . These updates are essential for tracking the aggregated properties of data points within each grid, enabling the algorithm to compute meaningful statistics for clustering.

4.7 Discovering Knowledge Via Privacy-Preserving Streaming Truth Discovery

To enable privacy-preserving streaming truth discovery in a scalable and cost-effective manner, our design leverages a network of satellites functioning as streaming servers and employs a lightweight cryptographic primitive, specifically secret sharing, to protect the privacy of each DP data stream sensing values. At each epoch, given a sensing value x , the DP splits it into several shares $\{[x]_0, [x]_1, \dots, [x]_n\}$ by randomly choosing $\{[x]_0, [x]_1, \dots, [x]_{n-1}\} \in F_q$ and computing $[x]_n = x - ([x]_0 + [x]_1 + \dots + [x]_{n-1}) \in F_q$, where F is a finite field and q is a large prime. We denote this additive secret-sharing scheme as $SS(\cdot)$. For demonstration purposes, we assume the data share $[x]_i$ is later securely given to the satellite S_i . To simplify the mathematics and enhance system understanding, we split the x into two shares $[x]_0$ and $[x]_1$, by randomly choosing $[x]_0 \in F_q$ and computing $[x]_1 = x - [x]_0 \in F_q$. The satellites then jointly perform a secure streaming truth discovery algorithm over the shares to discover truthful knowledge. We start by describing how to securely perform the atomic operations (i.e., secure addition, multiplication, division, and logarithm) underlying the streaming truth discovery algorithm, based on the shares of DPs' sensing values.

Algorithm 1 D-stream algorithm with data weights

```

Timec = 0;
Initialize an empty hash table grid_list;
while data stream is active do
    Capture data point in the data stream  $x = (x_1, x_2, \dots, x_d)$ ;
    Capture the weight of the data point  $w$ ;
    Determine the density grid  $g$  that contains  $x$ ;
    if  $g$  not in grid_list then
        Insert  $g$  into grid_list;
        Initialize the characteristic vector of  $g$ ;
    end if
    // Update the characteristic vector of  $g$  with  $x$  and  $w$ ;
     $g.density \leftarrow g.density + w$ ;
    for  $i = 1$  to  $d$  do
        //  $d$  = data points dimensionality;
         $g.sum\_coordinates[i] \leftarrow g.sum\_coordinates[i] + x[i] * w$ ;
         $g.sum\_squares[i] \leftarrow g.sum\_squares[i] + (x[i]^2) * w$ ;
    end for
    if Timec == gap then
        // Initial_clustering(grid_list);
        foreach grid  $g$  in grid_list do
            if  $g.density$  exceeds a predefined threshold then
                Assign  $g$  to a cluster;
            else
                Mark  $g$  as noise;
            end if
        end foreach
    end if
    if Timec mod gap == 0 then
        Detect and remove sporadic grids from grid_list;
        // Adjust_clustering(grid_list);
        foreach grid  $g$  in grid_list do
            // Update  $g.density$  based on decay function;
            if  $g.density$  falls below a predefined threshold then
                Remove  $g$  from the cluster and mark as noise;
            else
                Reassign  $g$  to the most appropriate cluster;
            end if
        end foreach
    end if
    Timec++;
end while

```

To begin with, we will consider how to perform secure addition and multiplication. The goal here is to add or multiply the shares of two inputs, e.g., $[x]_i$ and $[y]_i$, so as to allow satellite S_i (where $i \in \{0, 1\}$) to get a share $[x + y]_i$ or $[x \cdot y]_i$ without leaking any information about the private inputs x and y to both satellites. For the addition operation, it is straightforward: S_i computes $[x + y]_i = [x]_i + [y]_i$. For multiplication, if one of the inputs is a constant $A \in F_q$, S_i computes $[A y]_i = A[y]_i$. We denote multiplication by a constant as $Mul_c([y]_i, A)$. If both inputs are shared values, a viable technique for multiplication should be explored. In particular, our design adapts Beaver's multiplication triplet technique [30, 31] to support multiplication over secret-shared values. Suppose that the satellites already share a multiplication triplet $(a, b, c) \in F_q^3$, where a and b are random values and $a \cdot b = c \in F_q$. Beaver's technique implies that multiplication over shared values x and y can be performed as follows: each S_i locally computes $[u]_i = [x]_i - [a]_i$ and $[v]_i = [y]_i - [b]_i$. Then, each S_i broadcasts $[u]_i$ and $[v]_i$. Now, each S_i can reconstruct u and v , and further compute the share $[x \cdot y]_i = u \cdot v / 2 + u \cdot [b]_i + v \cdot [a]_i + [c]_i$.

The current blockchain network is not scalable enough to handle complex cryptographic encryption processes. Thus, for high cost efficiency [32], our design instead asks the DP and satellites to generate the multiplication triplets, thereby avoiding the expensive overhead associated with multiplication triplet generation.

Beyond secure addition and multiplication, the streaming truth discovery algorithm also necessitates secure division and logarithm operations. These operations are challenging to manage directly with secret sharing techniques alone. Thus, we employ garbled circuits to securely reconstruct original values, execute the required division and/or logarithm operations, and then secret-share the computation result between the two satellites for subsequent processing. For example purpose, our approach will involve S_0 (acting as the garbler) providing S_1 (acting as the evaluator) with a garbled circuit that combines their shares. The division and/or logarithm computation is performed on the recovered data. To securely share the computation result between the two satellites, the garbled circuit evaluation on S_1 's side outputs a masked result $m - r \in F_q$, where m is the computation result and r is a random value chosen by S_0 . Consequently, S_1 obtains $[m]_1 = m - r$ as its share of m , while S_0 obtains the corresponding share $[m]_0 = r$.

Non-linear functions such as logarithm and exponentiation are challenging to compute efficiently within garbled circuits [33]. To mitigate this issue, we aim to find a secure computation-friendly approximation for the logarithm function, ensuring its efficient support in the encrypted domain. Although it is widely acknowledged that such functions can be approximated by high-degree polynomials, e.g., Taylor series expansion, it is inefficient to handle high-degree polynomials in the encrypted domain [34]. Instead, we utilize efficient approximation using low-degree polynomials, particularly the technique of piecewise linear polynomial approximation [33]. Specifically, we approximate the non-linear logarithm function using a set of piecewise continuous polynomials with a polynomial degree of 1. Consequently, the logarithm function within the garbled circuit can be replaced by relatively lightweight operations such as comparison, addition, and (depth-1) multiplication.

4.7.1 Submission of Encrypted Sensory Values

During epoch l , each DP k splits their sensing value x_l^k into two shares $([x_l^k]_0, [x_l^k]_1)$ through the use of a standard additive secret sharing method $SS(\cdot)$. Besides encrypting the sensing value, each DP k also randomly generates two sets of multiplication triplets $Tri_0 = (a_0, b_0, c_0)$ and $Tri_1 = (a_1, b_1, c_1)$, and then divides each set into two shares using $SS(\cdot)$. The

resulting shares $\{[x_l^k]_i, [Tri_0]_i, [Tri_1]_i\}$ are then securely transmitted to the satellite S_i through an authenticated and secure channel.

4.7.2 Secure Truth Update at Satellites

In each epoch l , satellite S_i updates its share of the truth, denoted as Φ_i , to blockchain through weighted aggregation. During the initial epoch ($l = 1$), where the weights w_k for each DP k are initialized as constants, each satellite S_i independently computes the truth share Φ_i using Eq. (21) in Algorithm 2. In subsequent epochs ($l \geq 2$), however, satellite S_i only possesses a share of the weight $[w_k]_i$ and sensing value $[x_l^k]_i$ from each DP k . To compute a share of the multiplication result $[w_k \cdot x_l^k]_i$, Mul_{tri} is invoked, utilizing the triplet Tri_0 provided by each DP for this operation. For the division operation, GC_{div} protocol is employed to update the truth Φ securely from shares of the weighted sum of sensing values $([t]_0, [t]_1)$ and shares of the weights $([z]_0, [z]_1)$, as outlined in Eqs. (22) and (23) of Algorithm 2. The GC_{div} protocol proceeds as follows:

- The process begins with S_0 constructing a garbled circuit C_{div} , designed to accept as inputs the garbled values of $[t]_0, [t]_1, [z]_0, [z]_1$, and a randomly selected mask value r . Subsequently, S_0 transmits both the garbled circuit and the encrypted values $([t]_0, [z]_0, \tilde{r})$ to S_1 .
- Then, S_1 executes an oblivious transfer protocol with S_0 to acquire its respective garbled values $\tilde{[t]}_1$ and $\tilde{[z]}_1$. With the received encrypted data $[t]_0, [z]_0, [t]_1, [z]_1$, and $\tilde{r}$, S_1 evaluates the garbled circuit C_{div} , and get the output of $\Phi - r$. This output represents the truth share Φ_1 computed on the S_1 side.

4.7.3 Secure Weight Update at Shard

Upon blockchain and satellite nodes receive the updated truth share Φ_i , each satellite S_i proceeds to adjust the weight of each DP k to get an updated weight share $[w_i]$. Initially, the squared error $d_k = (x_l^k - \Phi)^2$ must be computed for each DP k in the encrypted domain using Eq. (24) in Algorithm 2, where the inputs are $[d]_i = ([x_l^k]_i - \Phi_i)$ and the second triplet Tri_1 is employed. Subsequently, the accumulated distance $st(k)$ for each DP k is updated via $[st(k)]_i \cdot \lambda + [d_k]_i$, with $\lambda \in [0, 1]$ serving as a decay rate to modulate the influence of historical sensing values on the current weight update. Using $[st(k)]_i$ and the total accumulated distances $[st^*]_i$, the weight for each DP k is adjusted through the garbled circuit protocol $GC_{div+log}$, as elucidated in Eq. (25) of Algorithm 2. The protocol $GC_{div+log}$ operates akin to the secure truth update phase, wherein within $GC_{div+log}$, shares are initially combined to retrieve the original values, followed

Algorithm 2 Privacy-preserving streaming truth discovery scheme for continuously mining reliable knowledge

Secure truth update ($\{([x_l^k]_i, [w_k]_i, [Tri_0]_i)_{i \in \{0,1\}}\}_{k=1}^K$):

At epoch $l = 1$:

1. S_i initializes $w_k = 1$ and sets the accumulated distance (i.e., state) $[st(k)]_i = 0$ for all DP k .
2. Upon receiving the shares of sensing values $\{[x_l^k]_i\}_{k=1}^K$ from DPs, S_i computes a share of truth Φ_i using the formula:

$$\Phi_i \leftarrow \sum_{k=1}^K Mul_c([x_l^k]_i, w_k) / \sum_{k=1}^K w_k \quad (21)$$

At epochs ($l \geq 2$)

3. Upon receiving the shares of sensing values $\{[x_l^k]_i\}_{k=1}^K$ from DPs, S_i computes a share of weighted sum t using the shares of DP weights $\{[w_k]_i\}_{k=1}^K$ through

$$[t]_i \leftarrow \sum_{k=1}^K Mul_{tri}([x_l^k]_i, [w_k]_i) \quad (22)$$

// Using the triplet $[Tri_0]_i$ to ensure secure multiplication

4. To obtain a share of truth Φ_i using t_i and $\{[x_l^k]_i\}_{k=1}^K$, S_i first we compute $[z]_i \leftarrow \sum_{k=1}^K [w_k]_i$, then we utilize the GC functionality GC_{div} for secure division

$$\Phi_i \leftarrow GC_{div}([t]_i, [z]_i) \quad (23)$$

// $\Phi_i = ([t]_0 + [t]_1) / \sum_{k=1}^K ([w_k]_0 + [w_k]_1)$

Secure weight update ($\{\Phi_i, ([x_l^k]_i, [st(k)]_i, [Tri_1]_i)_{i \in \{0,1\}}\}_{k=1}^K$):

At epoch $l \geq 1$:

For each DP k :

5. After acquiring the updated truth share Φ_i , S_i calculates $[d]_i \leftarrow ([x_l^k]_i - \Phi_i)$.
6. S_i calculates the share of the squared error $[(x_l^k - \Phi_i)^2]_i$ through

$$[d_k]_i \leftarrow Mul_{tri}([d]_i, [d]_i) \quad (24)$$

// Using the triplet $[Tri_1]_i$ to ensure secure multiplication

7. S_i updates the state of DP k by computing $[st(k)]_i \leftarrow [st(k)]_i \cdot \lambda + [d_k]_i$.
8. S_i aggregates all states to compute $[st^*]_i \leftarrow \sum_{k=1}^K [st(k)]_i$.
9. S_i updates the weight share $[W_k]_i$ by employing the GC functionality $GC_{div+log}$, which securely performs division and logarithm operations on the shares $[st(k)]_i$ and $[st^*]_i$, ensuring the secure update of $[W_k]_i$:

$$[w_k]_i \leftarrow GC_{div+log}([st(k)]_i, [st^*]_i) \quad (25)$$

// $w_k = -\log([st(k)]_0 + [st(k)]_1) / ([st^*]_0 + [st^*]_1)$

by performing division and logarithm operations on these values to produce a masked result. Each satellite subsequently obtains a secret-shared result. The detailed execution protocol unfolds as follows:

- S_0 begins by constructing a garbled circuit $\mathcal{C}_{div+log}$, which accepts as inputs the garbled values of $[st(k)]_0, [st(k)]_1, [st^*]_0, [st^*]_1$, along with a randomly chosen mask value r^* . Subsequently, S_0 transmits both the garbled circuit and the encrypted values $([st(k)]_0, [st^*]_0, \tilde{r}^*)$ to S_1 .
- Then, S_1 executes an oblivious transfer protocol with S_0 to acquire its respective garbled values $[st(k)]_1$ and $[st^*]_1$. Utilizing the received encrypted data $[st(k)]_0, [st^*]_0, [st(k)]_1, [st^*]_1$, and $\tilde{r}^*$, S_1 evaluates the

garbled circuit $\mathcal{C}_{div+log}$, getting an output of $w_k - r^*$. The result will be the weight share $[w_k]_1$ computed on the S_1 side.

4.7.4 Security Guarantees

OrbitalChain framework ensures robust privacy protection under the assumption of non-colluding and semi-honest adversaries during the execution of secure streaming truth discovery. The security of our approach can be proved and assured by a thorough examination of its operational flow, which can be formally substantiated through the modular sequential composition theorem [35], [36].

In each epoch, each satellite receives a share of all DPs' sensing values along with a share of the multiplication triplets. The robustness of the secret-sharing technique ensures complete confidentiality, preventing any information leakage to either satellite. For truth and weight updates, the satellites operate on their respective secret-shared values locally and engage in secure interactions with each other. These interactions leverage Beaver's multiplication technique for secure multiplication and garbled circuits for secure division and logarithm calculations. The inherent security properties of Beaver's technique [30] and garbled circuits [37] ensure that no sensitive information is disclosed during these interactions. Consequently, each satellite only obtains a share of the computation result, maintaining the overall security of the system. By integrating these advanced cryptographic techniques into the OrbitalChain framework, we establish a zero-trust environment where every transaction and computation is conducted with minimal reliance on trust assumptions. This approach not only mitigates the risks associated with potential adversarial behavior but also enhances the system's resilience against data breaches and unauthorized access.

4.8 Consensus Mechanism in the Satellite Network

In our satellite-based sharded blockchain network, the real-time data stream transactions are initially conducted off-chain. Each shard, functioning independently within the network, processes its own transactions off-chain to ensure swift and efficient handling of data streams. This method leverages the inherent advantages of sharding by distributing the transaction workload across multiple shards, thus significantly reducing latency, enhancing the throughput of the network and optimizing the performance and scalability of the blockchain network. After completing off-chain transactions within a shard, to elevate system security, each shard initiates an inter-shard transaction verification process (fig. 4). This step ensures that the transaction is cross-validated by other shards

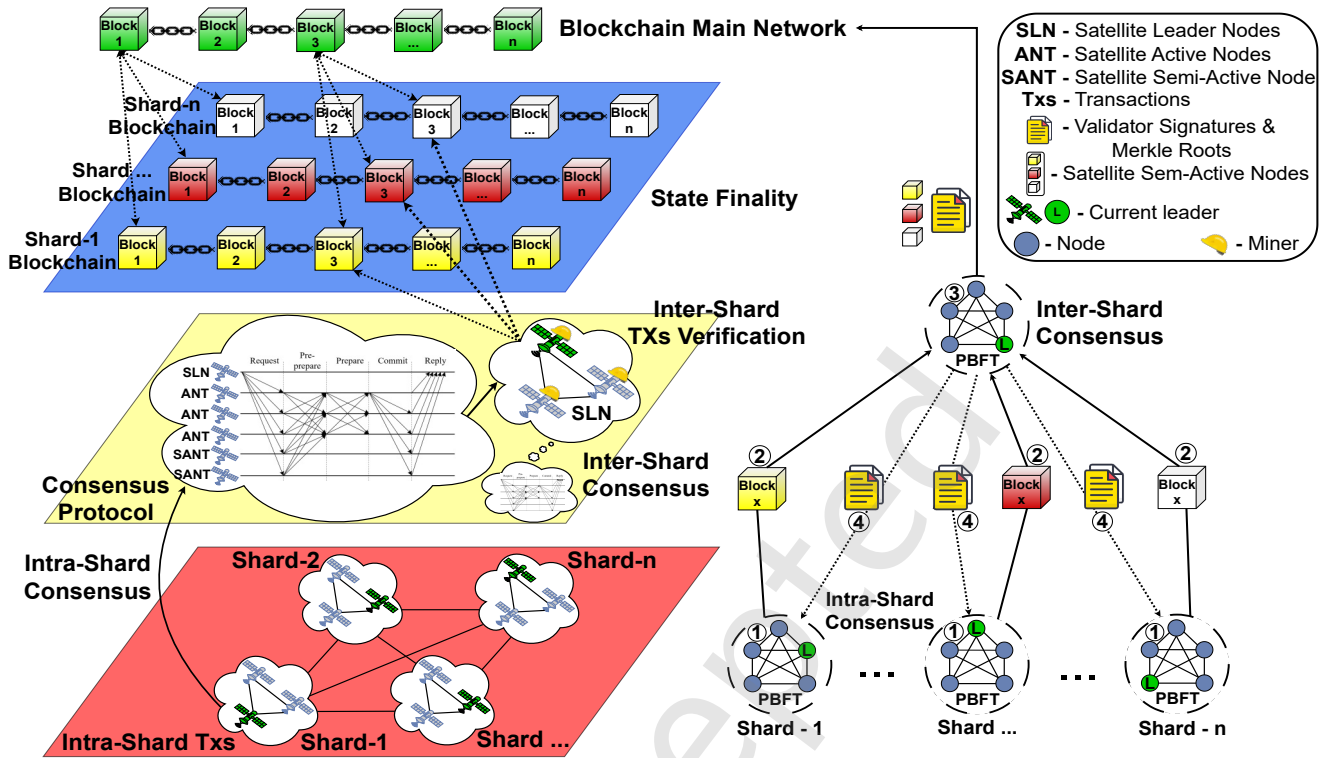


Fig. 4: Transaction process in OrbitalChain

within the network. By requesting verification from multiple shards, the system enhances its integrity and trustworthiness. Once the inter-shard verification is successfully completed, the transaction is input into the main blockchain system for final validation and permanent recording. The inter-shard transaction verification process depicted in figure 5 utilizes a PBFT [38, 39] protocol to achieve consensus across multiple shards in a blockchain network.

The Random leader at time T_x (RT_x Leader) coordinates the process by managing communication between shard leaders and ensuring efficient communication, consensus among shard leaders and that each stage is completed successfully:

Request: The Request stage begins when a shard x wants to validate a set of transactions that was initially made off-chain. These transactions need to be verified by other shards before it can be added to the main blockchain. The shard leader of Shard x generates a transaction request containing the transactions details and sends it to the RT_x Leader, who will coordinate the validation process across multiple shards.

Pre-Prepare: In the Pre-Prepare stage, the RT_x Leader receives the transaction request from Shard x and disseminates the transaction set details to all other shard leaders involved in the validation process. The RT_x Leader creates a Pre-Prepare message, which includes the transaction data and a unique identifier for tracking the transaction throughout

the consensus process. This message is sent to the leaders of all relevant shards. This dissemination ensures that all shard leaders are aware of the transaction and can begin the necessary preparations for verification.

Intra-Shard Consensus: The Intra-Shard Consensus stage involves each shard independently validating the transactions sent by Shard x . Upon receiving the Pre-Prepare message, each shard leader initiates an intra-shard consensus process. This process is akin to the Smart Byzantine Fault Tolerance (SBFT) [40] protocol and involves multiple nodes within each shard validating the transaction. The intra-shard consensus process includes the following sub-stages [41]:

- **Pre-Prepare:** The shard leader distributes the transaction details to all nodes within the shard.
- **Prepare:** Shard nodes validate the transactions and exchange Prepare messages with each other to confirm their individual validation results.
- **Commit:** Once a sufficient number of Prepare messages are received, shard nodes send Commit messages, indicating agreement on the transactions validity.
- **Reply:** After achieving consensus, each shard node sends a Reply message to the shard leader. The shard leader then compiles these replies to finalize the validation decision within the shard.

This intra-shard consensus ensures that each shard inde-

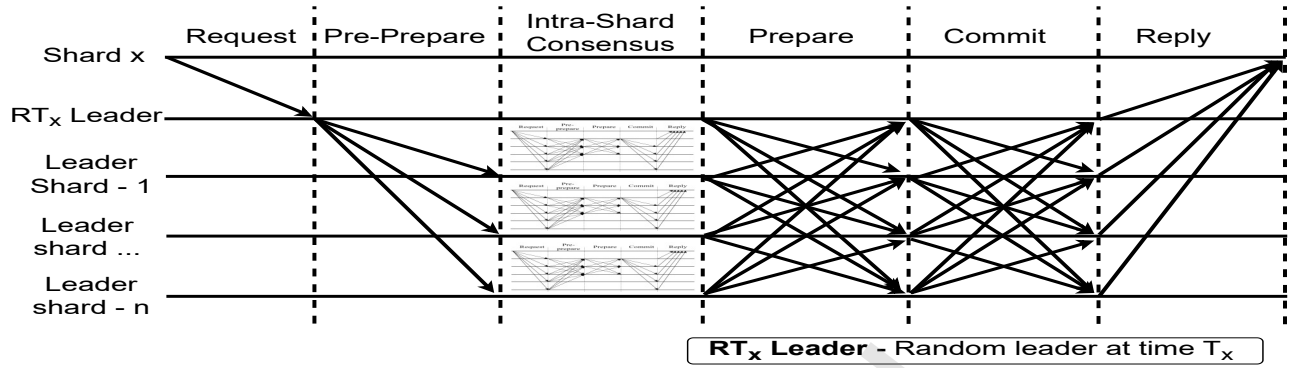


Fig. 5: Inter-shard transactions verification

pendently verifies the transaction before contributing to the overall inter-shard consensus.

Prepare: Following the successful intra-shard consensus, the Prepare stage involves gathering preliminary commitments from all shard leaders. Each shard leader, having reached consensus within their shard, sends a Prepare message to the RT_x Leader. These Prepare messages indicate that the shards have validated the transactions and are ready to proceed. The RT_x Leader collects these preliminary agreements, ensuring that all involved shards are aligned before moving forward.

Commit: In the Commit stage, the RT_x Leader collects Prepare messages from all involved shard leaders. Once the RT_x Leader has received enough Prepare messages to meet the consensus threshold, a Commit message is sent back to the shard leaders. This Commit message serves as the final confirmation that the transactions are agreed upon and should be recorded on the main blockchain. The Commit stage ensures that all shard leaders are synchronized and ready to finalize the transaction, thus maintaining the integrity and consistency of the blockchain across all shards.

Reply: The Reply stage concludes the inter-shard transaction verification process by confirming the transaction completion to Shard x . After receiving the Commit message, each shard leader sends a Reply message back to the RT_x Leader, confirming that the transaction has been committed within their shard. The RT_x Leader compiles these confirmations and sends a final Reply message to Shard x . This final Reply indicates that the transaction has been successfully verified and committed across all involved shards, and it can now be added to the main blockchain. This multi-stage process maintains the integrity and consistency of the blockchain, providing a reliable mechanism for handling complex transactions in a distributed and decentralized network. The RT_x Leader is chosen randomly from a pool of shard leaders at a determinate time x , ensuring a fair and unbiased selection

process. Leaders with higher reliability scores, reflecting their consistent performance and trustworthiness within the network, are weighted more heavily in the selection process. This hybrid approach ensures that the RT_x Leader is both randomly selected and highly reliable, enhancing the overall robustness and efficiency of the network's transaction verification process.

5 Formal Security Analysis

5.1 Security of Secret Sharing and Multiplication

To develop our formal security analysis for our secret-sharing scheme, we based on principles from the work [42]. Their work offers a privacy-preserving approach for securely aggregating information, which aligns with the goals of our multi-value secret sharing scheme in a distributed setting.

Correctness 5.1. The of the security scheme is guaranteed when all n parties' shares are collected. Given the shares $[x_j]_i$ for each value x_j , the secret can be reconstructed as:

$$x_j = \sum_{i=1}^n [x_j]_i \mod q \quad (26)$$

This ensures that the reconstructed value x_j is equal to the original secret.

Theorem 5.1. The scheme is secure against an adversary that corrupts fewer than t parties. That is, given fewer than t shares, no information about the original secret values can be learned.

For each secret x_j , the shares $[x_j]_1, [x_j]_2, \dots, [x_j]_n$ are uniformly distributed random elements in Z_q . Each share is generated independently, and the sum of the shares is constrained to reconstruct x_j . However, if fewer than t shares are known, the adversary can only observe random elements from Z_q and cannot derive any useful information about x_j .

Let S denote a subset of up to t corrupted parties. The joint distribution of the shares in S is independent of x_j , meaning:

$$H(x_j | [x_j]_i \text{ for } i \in S) = H(x_j). \quad (27)$$

Thus, the adversary learns nothing about x_j from observing fewer than t shares, ensuring perfect privacy.

Theorem 5.2. The product of two secret-shared values can be securely computed using Beaver's multiplication triples without revealing any information about the original values.

Let x and y be two secret-shared values, and let a, b, c be pre-shared random values such that $a \cdot b = c$. Using Beaver's triples, the product $z = x \cdot y$ is computed securely as:

$$z = u \cdot v + u \cdot [b]_i + v \cdot [a]_i + [c]_i, \quad (28)$$

where $u = x - [a]_i$ and $v = y - [b]_i$. The randomization introduced by a and b ensures that the shares of the product z reveal no information about x or y . The adversary cannot infer any useful information from the shares of u or v since they are independent of x and y , preserving the security of the multiplication.

5.2 Game-Based Security Proof for the Privacy-preserving streaming truth discovery protocol

We provide a formal security analysis of Algorithm 2 using the simulation-based paradigm introduced in Section 4.3. We prove that the protocol securely realizes an ideal functionality $\mathcal{F}_{\text{truth}}$ against semi-honest adversaries (Definition 4.1) through a sequence of computationally indistinguishable games.

We formally specify the ideal functionality that captures the security guarantees of Algorithm 2.

Definition 5.1. The ideal functionality $\mathcal{F}_{\text{truth}}$ interacts with satellites $\{S_0, S_1\}$, data providers $\{DP_1, \dots, DP_K\}$, and simulator $\mathcal{S}$ as follows:

Input Phase:

- (1) Upon receiving (INPUT, sid, x_l^k) from data provider DP_k at epoch l , store x_l^k and send (RECEIPT, sid, DP_k) to $\mathcal{S}$.
- (2) For corrupted $S_i \in \mathcal{C}$, $\mathcal{S}$ may observe input shares $[x_l^k]_i$ for all $k \in [K]$.

Computation Phase:

- (3) Once all inputs $\{x_l^k\}_{k=1}^K$ are received, compute:
 - Truth value: $\Phi_l = \sum_{k=1}^K w_k \cdot x_l^k / \sum_{k=1}^K w_k$
 - Squared errors: $d_k = (x_l^k - \Phi_l)^2$ for each k
 - Updated weights: $w_k = -\log \left(st(k) / \sum_{j=1}^K st(j) \right)$

where $st(k) = \lambda \cdot st(k) + d_k$ is the accumulated distance with decay factor λ .

Output Phase:

- (4) Send (OUTPUT, sid, $[\Phi_l]_i, \{[w_k]_i\}_{k=1}^K$) to satellite S_i for $i \in \{0, 1\}$.
- (5) Send $\{[\Phi_l]_i, [w_k]_i\}_{i \in \mathcal{C}}$ to $\mathcal{S}$ (output shares of corrupted satellites).

The functionality $\mathcal{F}_{\text{truth}}$ guarantees: (i) Input Privacy - no information about individual sensing values x_l^k leaks beyond the computed truth Φ_l ; (ii) Correctness - outputs are computed according to the streaming truth discovery algorithm; (iii) Independence - honest inputs remain independent of corrupted parties' views.

5.2.1 Simulator Construction.

We construct a simulator $\mathcal{S}$ that produces a view indistinguishable from the real protocol view (Definition 4.2) while interacting only with $\mathcal{F}_{\text{truth}}$.

Definition 5.2. For semi-honest adversary $\mathcal{A}$ corrupting satellite S_0 (the case for S_1 is symmetric), simulator $\mathcal{S}$ proceeds as follows:

Step 1 (Setup): $\mathcal{S}$ receives security parameter 1^κ , corrupted set $\mathcal{C} = \{S_0\}$, and input shares $\{[x_l^k]_0, [Tri]_0, [Tri]_0\}_{k=1}^K$ from all data providers.

Step 2 (Input Simulation): For each honest satellite S_1 :

- Sample simulated shares $[x_l^k]_1 \leftarrow \mathbb{F}_q$ uniformly at random
- Sample simulated Beaver triples $[\widetilde{Tri}]_1, [\widetilde{Tri}]_1 \leftarrow \mathbb{F}_q^3$
- Sample simulated randomness $\tilde{r}_1 \leftarrow \{0, 1\}^{\text{poly}(\kappa)}$

Step 3 (Protocol Message Simulation): For each protocol phase:

- *Beaver multiplication:* Simulate (u, v) pairs by sampling $\tilde{u}, \tilde{v} \leftarrow \mathbb{F}_q$
- *Garbled circuits $GC_{\text{div}}, GC_{\text{div}+\log}$:* As garbler, S_0 constructs circuits normally; simulator uses standard GC simulation to produce indistinguishable circuit views

Step 4 (Output Retrieval): $\mathcal{S}$ receives (OUTPUT, sid, $[\Phi_l]_0, \{[w_k]_0\}_{k=1}^K$) from $\mathcal{F}_{\text{truth}}$ and outputs the simulated view:

$$\widetilde{\text{View}}_0 = \left(\{[x_l^k]_0\}_{k=1}^K, r_0, \{\tilde{u}, \tilde{v}\}, \widetilde{GC}, [\Phi_l]_0, \{[w_k]_0\}_{k=1}^K \right) \quad (29)$$

5.2.2 Protocol Security Analysis

Theorem 5.3. Under the semantic security of the encryption scheme and the hiding property of the commitment scheme, Algorithm 2 securely realizes $\mathcal{F}_{\text{truth}}$ in the semi-honest model.

Formally, for any PPT adversary $\mathcal{A}$ corrupting at most one satellite, there exists a PPT simulator $\mathcal{S}$ such that:

$$\{\text{IDEAL}_{\mathcal{F}_{\text{truth}}, \mathcal{S}}(1^\kappa, \vec{x})\} \stackrel{c}{=} \{\text{REAL}_{\pi, \mathcal{A}}(1^\kappa, \vec{x})\} \quad (30)$$

We prove security through a sequence of games, where each consecutive pair is computationally indistinguishable. Without loss of generality, we prove for corrupted S_0 ; the proof for S_1 is symmetric.

Game $\mathbf{G}_0$: This is the real protocol execution where $\mathcal{A}$ receives the view of S_0 :

$$\begin{aligned} \mathbf{G}_0 &= \text{View}_0^\pi(\vec{x}) \\ &= (\{[x_i^k]_0, [\text{Tr}_i]_0\}_{k=1}^K, r_0, \{(u, v)\}, GC, [\Phi_l]_0, \{[w_k]_0\}) \end{aligned} \quad (31)$$

Game $\mathbf{G}_1$: Identical to $\mathbf{G}_0$, except we replace honest satellite S_1 's input shares with uniformly random values: $[x_i^k]_1 \Rightarrow [x_i^k]_1 \leftarrow \mathbb{F}_q$.

Lemma 1. $\mathbf{G}_0 \equiv \mathbf{G}_1$ (perfectly indistinguishable).

By the perfect secrecy of additive secret sharing over $\mathbb{F}_q$, each share $[x_i^k]_0$ is uniformly distributed and independent of the secret x_i^k . The joint distribution of S_0 's shares is identical whether S_1 holds real shares or random values, since S_0 never observes S_1 's shares directly.

Game $\mathbf{G}_2$: Identical to $\mathbf{G}_1$, except we replace the Beaver multiplication transcript with simulated values.

Lemma 2. $|\Pr[\mathcal{D}(\mathbf{G}_1) = 1] - \Pr[\mathcal{D}(\mathbf{G}_2) = 1]| \leq K \cdot \epsilon_{\text{Mult}}(\kappa)$

In Beaver's protocol, S_0 observes (u, v) where $u = x - a$ and $v = y - b$ for random pre-shared values $a, b \in \mathbb{F}_q$. Since a and b are uniformly random and independent of x and y , the values (u, v) are also uniformly random.

Suppose distinguisher $\mathcal{D}$ distinguishes with advantage ϵ . We construct reduction $\mathcal{B}$ against Beaver triple security: $\mathcal{B}$ embeds the challenge in one multiplication instance and simulates the rest. If the challenge uses real inputs, the view is $\mathbf{G}_1$; if random, the view is $\mathbf{G}_2$. By [30], $\epsilon_{\text{Mult}}(\kappa) = \text{negl}(\kappa)$. With K multiplications per epoch, the union bound gives the stated advantage.

Game $\mathbf{G}_3$: Identical to $\mathbf{G}_2$, except we replace garbled circuit executions with simulated circuits.

Lemma 3. $|\Pr[\mathcal{D}(\mathbf{G}_2) = 1] - \Pr[\mathcal{D}(\mathbf{G}_3) = 1]| \leq 2 \cdot \epsilon_{GC}(\kappa)$

For GC_{div} and $GC_{\text{div}+\log}$, satellite S_0 acts as the garbler and constructs the circuit. By garbled circuit security [37], the garbler learns only its output share $[\Phi_l]_0 = r$ (the mask it chose). The simulator constructs a simulated circuit given only

the output, which is computationally indistinguishable from the real garbled circuit. With two garbled circuit invocations (GC_{div} for truth update and $GC_{\text{div}+\log}$ for weight update), the total advantage is $2 \cdot \epsilon_{GC}(\kappa)$.

Game $\mathbf{G}_4$: This is the ideal world simulation where $\mathcal{S}$ interacts with $\mathcal{F}_{\text{truth}}$. The simulator:

- Receives corrupted input shares from S_0
- Simulates all honest party messages using random values (as in $\mathbf{G}_3$)
- Obtains output shares $[\Phi_l]_0, \{[w_k]_0\}$ from $\mathcal{F}_{\text{truth}}$
- Outputs the assembled view View_0

Lemma 4. $\mathbf{G}_3 \equiv \mathbf{G}_4$ (perfectly indistinguishable).

In $\mathbf{G}_3$, outputs $\{[\Phi_l]_0, [w_k]_0\}$ are computed by executing the protocol with simulated inputs. In $\mathbf{G}_4$, outputs are obtained from $\mathcal{F}_{\text{truth}}$. By the correctness of $\mathcal{F}_{\text{truth}}$ (Definition 5.1), these outputs are identically distributed. All other view components are constructed identically in both games.

We combine the games with the triangle inequality:

$$\begin{aligned} &|\Pr[\mathcal{D}(\text{REAL}_{\pi, \mathcal{A}}) = 1] - \Pr[\mathcal{D}(\text{IDEAL}_{\mathcal{F}, \mathcal{S}}) = 1]| \\ &\leq \sum_{i=0}^3 |\Pr[\mathcal{D}(\mathbf{G}_i) = 1] - \Pr[\mathcal{D}(\mathbf{G}_{i+1}) = 1]| \\ &\leq 0 + K \cdot \epsilon_{\text{Mult}}(\kappa) + 2 \cdot \epsilon_{GC}(\kappa) + 0 = \text{negl}(\kappa) \end{aligned} \quad (32)$$

Since polynomially many invocations of negligibly-distinguishable hybrids remain negligible, security follows by the modular composition theorem [35].

Lemma 5. The protocol reveals to each satellite S_i only its output shares. No information about individual sensing values $\{x_i^k\}$ is leaked beyond what is computable from the truth Φ_l .

Each cryptographic component provides the following guarantees: (i) secret shares are uniformly random and independent of secrets; (ii) Beaver's protocol reveals only random masks (u, v) independent of inputs; (iii) garbled circuits reveal only designated outputs.

For state variables $st(k)$ maintained across epochs, S_i holds only shares $[st(k)]_i$, which remain uniformly random. The recurrence $st(k)_{l+1} = \lambda \cdot st(k)_l + d_k$ preserves the sharing structure, and weight computations occur entirely within garbled circuits. Thus, accumulated state does not leak additional information.

By simulation security, for any PPT adversary $\mathcal{A}$:

$$\Pr[\mathcal{A}(\text{View}_i^\pi) \text{ infers } x_i^k] \leq \Pr[\mathcal{A}(\Phi_l) \text{ infers } x_i^k] + \text{negl}(\kappa) \quad (33)$$

Since $\Phi_l = \sum_k w_k x_l^k / \sum_k w_k$ with $K \geq 2$ unknowns, the system is underdetermined, ensuring input privacy.

Lemma 6. *The protocol maintains security against any coalition of corrupted data providers, provided both satellites S_0 and S_1 remain honest and non-colluding.*

Corrupted data providers can only observe their own input values and the final published truth Φ_l . They do not receive any shares or protocol messages between satellites. The simulator construction extends naturally: $\mathcal{S}$ simulates the satellites' computation using the same techniques, and corrupted DP's views consist only of their inputs and published outputs, which are explicitly permitted by $\mathcal{F}_{\text{truth}}$.

The protocol achieves the following concrete security bound per epoch:

$$\text{Adv}^{\text{protocol}}(\kappa) \leq K \cdot \epsilon_{\text{Mult}}(\kappa) + 2 \cdot \epsilon_{GC}(\kappa) \quad (34)$$

where K is the number of data providers. Over L epochs, the total advantage is bounded by:

$$\text{Adv}^{\text{total}}(\kappa) \leq L \cdot (K \cdot \epsilon_{\text{Mult}}(\kappa) + 2 \cdot \epsilon_{GC}(\kappa)) = \text{negl}(\kappa) \quad (35)$$

Table 3: System specifications

PARAMETER	VALUE
GPU	4 NVIDIA RTX 3090 (24GB each)
CPU	Intel Xeon Silver4314 @2.40GHz (64 cores)
RAM	384GB DDR4
Blockchain Size	20 GB
Maximal Blocktime Delay	13.2s
Average Blocktime	0.233s
Average Transaction per second	14.8 TPS

6 Performance Analysis

6.1 Consensus Between Satellites

The simulations were conducted on our dedicated server equipped with an Intel Xeon Silver 4314 CPU featuring 64 cores and a base frequency of 2.40 GHz. The server is configured with 384 GB of DDR4 RAM and four NVIDIA GeForce RTX 3090 GPUs, each providing 24 GB of memory. The experiment was performed on a private, permissioned blockchain network designed to meet the specific requirements of our study. The detailed specifications of our system are summarized in Table 3.

PBFT protocols enhance network security by ensuring that consensus can be achieved even in the presence of malicious or faulty nodes, thereby providing resilience against various types of attacks such as replay, spoofing attacks and so on [41]. The system employs cryptography, including digital signatures, hashing, and message authentication codes (MACs), to protect against replay and spoofing attacks

As we demonstrated in the subsection 4.8, the sharding technique is to partition the satellite-based blockchain network into smaller and more manageable shards. Where each shard operates semi-independently. This approach allows for parallel processing of transactions, improving overall network throughput and scalability. Figure 6 analyzes the impact of the submitted data size on the computational power required by a satellite nodes to participate in the arbitration consensus process. The figure indicates that within individual shards, the number of nodes significantly affects computation costs, suggesting the need for careful optimization of shard size and inter-shard communication protocols. The larger block sizes and higher node counts increase computational overhead.

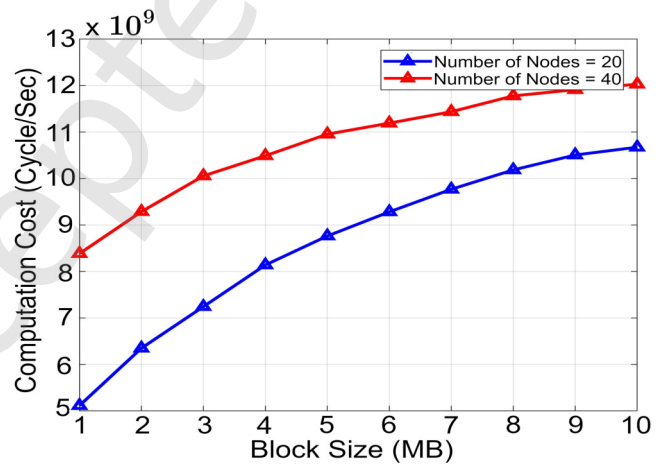


Fig. 6: Computing cost according to data size

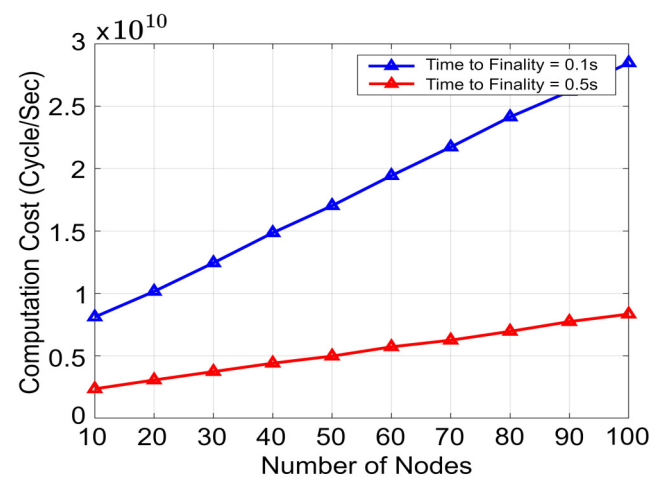


Fig. 7: Computing cost according to number of nodes

In figure 7, the results show that as group size increases, computational power requirements also rise. This increase reflects the growing complexity and overhead needed to main-

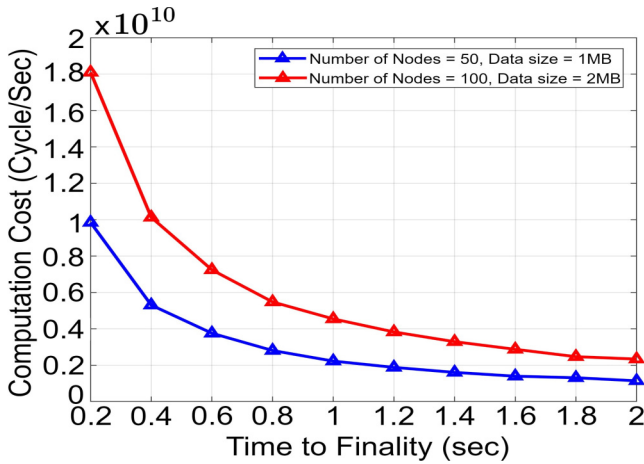


Fig. 8: Computing cost needed to finalize the process

tain consensus across a larger network. With more nodes, the protocol must handle more participants in the consensus process, increasing computational effort to ensure fault tolerance and network integrity. Satellites exchange and validate messages during the arbitration consensus process, so a larger group size means more messages and higher computing power demands. If the consensus process needs to be completed quickly, satellites must allocate more computing power to meet the requirement. The figure illustrates that the computation cost for a time to finality of 0.1 seconds is consistently higher than for 0.5 seconds due to stricter time constraints requiring more frequent consensus rounds. This emphasizes the need to optimize shard size and consensus protocols to balance computational costs and scalability. Adjusting the time to finality helps manage trade-offs between latency and computational efficiency. A shorter time to finality improves transaction throughput and responsiveness but increases computation, while a longer time to finality reduces computational demands but slows transaction confirmation. In sharding, the blockchain network is divided into smaller shards that operate semi-independently, contributing to improving the consensus performance.

Figure 8 demonstrates the relationship between computation cost and time to finality, which exhibits that the more time allocated, the less computing power is necessary to reach an agreement among the satellites. The observed decrease in computation cost with increasing time to finality is attributable to the reduced frequency of consensus rounds, allowing nodes more time to process and communicate. Consequently, a longer time to finality mitigates the computational load by distributing the consensus operations over a more extended period. Figure 8 also compares the effects of varying the number of satellites and the size for the submitted transactions data. Based on these observations, it would be

beneficial to set a small fixed size of satellites participating in the arbitration consensus process.

6.2 Satellites Weight Update

The data utilized in this study were sourced from the archival resources provided by CelesTrak (<https://celestrak.org/NORAD/archives/>) [43], a comprehensive repository of NORAD Two-Line Element Sets (TLEs). This dataset enabled the precise tracking and analysis of satellite trajectories, forming the basis of the experiments conducted to derive our results.

Figures 9, 10 and 11 illustrate the dynamic evolution of satellite weights within a decentralized data streaming system incorporating a blockchain-based truth discovery mechanism. This mechanism is essential for ensuring data integrity and mitigating the influence of malicious or faulty nodes. It iteratively refines the trustworthiness of each satellite within the network. Figure 9 focuses on a scenario with 10 satellites and provides a granular view of the weight adjustment process, where each epoch lasts 10 seconds, with truth discovery processing completing within 0.15–0.25 seconds. As we can see in the figure, satellite 5 stands out with a significantly higher initial weight, indicating a substantial initial influence. As the algorithm processes more data, particularly in the first few epochs (0 to 2.5), there are rapid fluctuations in weights, indicating aggressive adjustments. This phase is critical for the algorithm to quickly adapt to the characteristics of the incoming data stream.

After the initial turbulent phase, around epoch 2.5, the weights begin to stabilize, demonstrating the algorithm's transition to a steady state where it fine-tunes the weights. The convergence of weights around similar values suggests a balanced influence of the data points, reflecting effective normalization by the algorithm. The weights' minor fluctuations, even after stabilization, indicate continuous fine-tuning to maintain accuracy in response to new data. Satellite 2 and satellite 6 exhibit noticeable peaks around epochs 2 to 6, indicating periods of higher influence, which stabilize later, suggesting these satellites had transient but significant contributions. The smoother transitions in the weights of other satellites point to a consistent influence throughout the epochs.

From figures 9, 10 and 11 experiments, we decided to focus on discussing the results of figure 9 due to the fact that the approach focuses on a smaller number of satellites, and its visual presentation reduces complexity and enhance the ability to differentiate nuanced patterns and behaviours across epochs. This deliberate choice minimizes visual clutter, allowing for

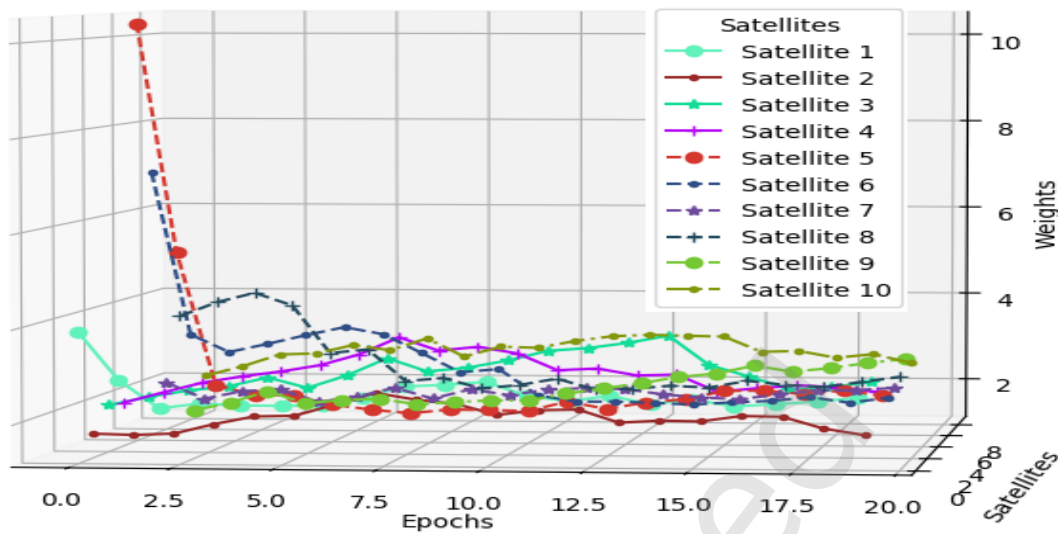


Fig. 9: 10 Satellites weight updates over epochs

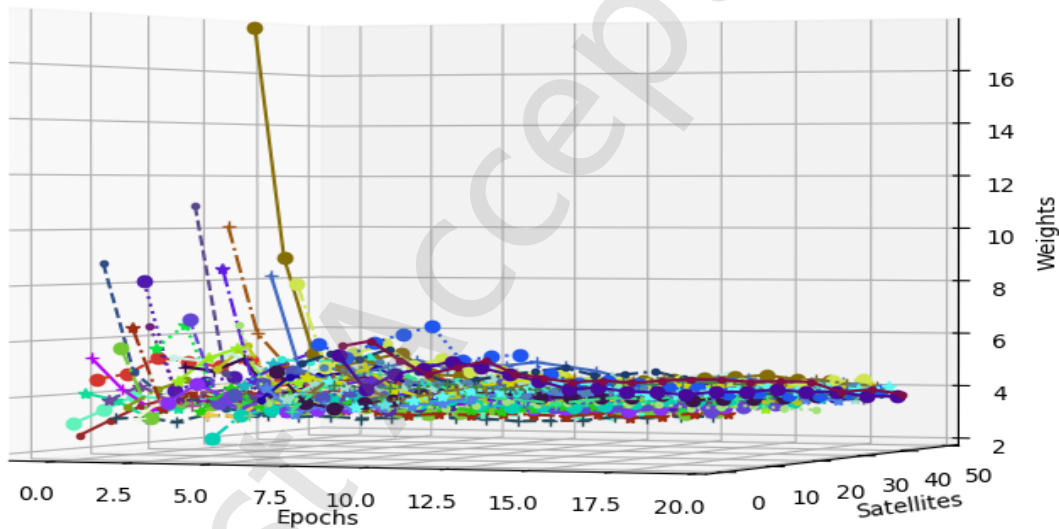


Fig. 10: 50 Satellites weight updates over epochs

a more precise examination of trends and interactions among the variables of interest. Due to the increasing number of satellites, we maintained the data collection window duration at 30 seconds, while the truth discovery processing time varied between 0.4–0.6 seconds per-epoch for the 50 satellite configuration and 0.6–0.8 seconds for the 100 satellite configuration, respectively. These measurements demonstrate that OrbitalChain maintains efficient and scalable performance under real-time orbital constraints.

Figure 12 provides an overview of the truth discovery algorithm's behavior, consolidating insights from figures 9, 10, and 11. By examining the algorithm's performance across different shard configurations (10, 50, and 100 satellites) over 20 epochs, this figure enables a comparative understanding of

how the algorithm adapts and converges under varying conditions. It highlights the impact of shard size on the stability and accuracy of the discovered truth, offering a clearer perspective on the algorithm's effectiveness in handling diverse satellite distributions. The truth for each epoch is computed iteratively, with weights being updated based on the squared deviation of each satellite's input from the discovered truth. The exponential decay factor λ is employed to lower past deviations. The analysis reveals distinct convergence behaviors for different values of shards. When shard has 10 satellites, the truth discovery process exhibits noticeable fluctuations throughout the epochs, indicating a higher sensitivity to individual satellite inputs. This is attributable to the smaller number of satellites, where each satellite's input significantly influences the aggre-

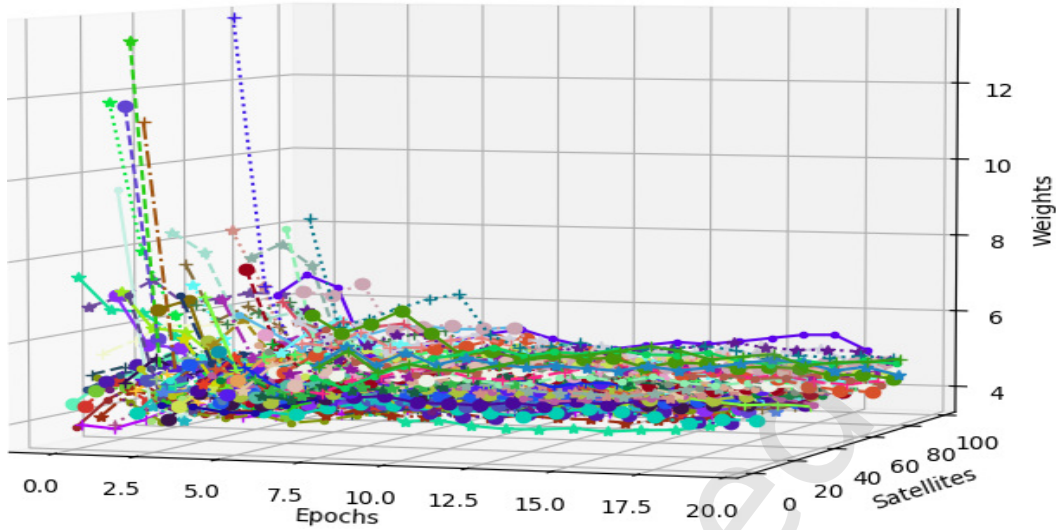


Fig. 11: 100 Satellites weight updates over epochs

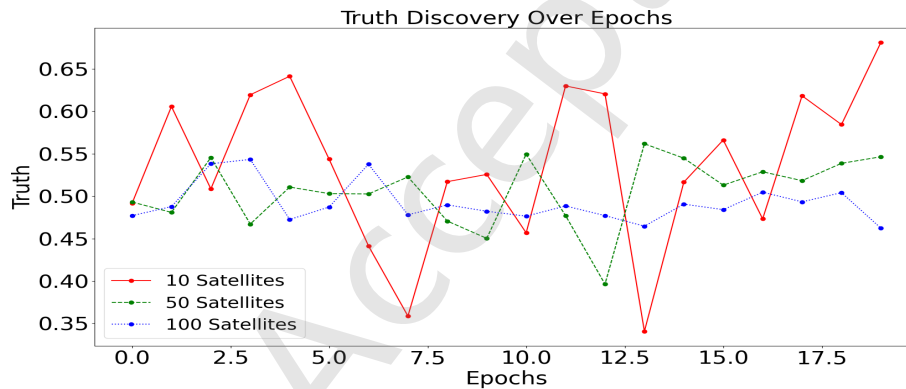


Fig. 12: Truth discovering over epochs

gated truth. The higher variability suggests that the algorithm requires more epochs to stabilize as it averages over fewer data points, leading to potential susceptibility to noise and outliers. As shard increases with 50 satellites, the convergence of the truth becomes more stable and less random. The increased number of satellites allows for a more robust aggregation of inputs, reducing the impact of any single satellite's deviation. Consequently, the truth discovery shows a smoother trajectory with fewer oscillations, reflecting a more reliable convergence pattern. This improvement in stability is due to the larger groups of satellites providing a more stable and accurate average. This means that the average of a larger sample is more likely to reflect the true value. For shard with 100 satellites, the convergence pattern further stabilizes, exhibiting the least amount of fluctuation among the three scenarios. The substantial increase in satellites ensures that the influence of individual deviations is minimized, and the aggregated truth converges rapidly and consistently. The results suggest that

with shard with 100 satellites, the algorithm achieves a high degree of reliability in truth discovery within the given 20 epochs. This scenario highlights the algorithm's effectiveness in leveraging a large number of satellites to derive a stable and accurate truth, demonstrating robustness against input variations and noise. As satellites in the shard increases, the algorithm's ability to converge to a stable truth improves significantly.

6.3 Data Clustering with D-Streaming Algorithm

Understanding the computational cost associated with different network configurations is crucial for optimizing the performance of Low Earth Orbit (LEO) satellite networks. In particular, the number of nodes functioning as streaming servers plays a significant role in determining the efficiency of data processing and distribution. This section examines how the computational cost varies with different numbers of nodes using the D-stream algorithm.

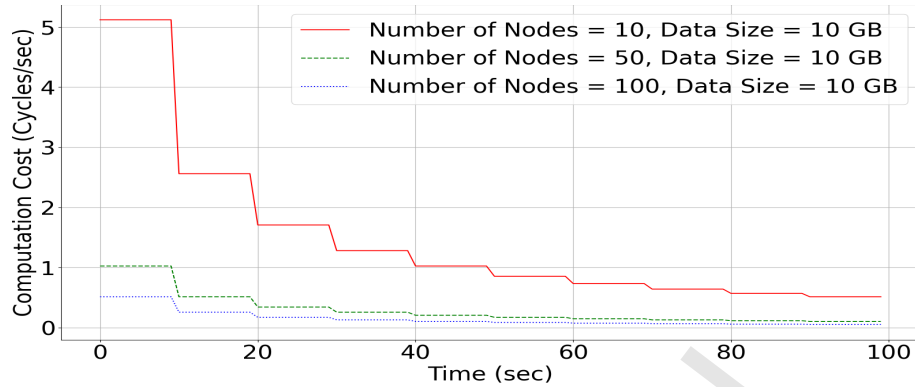


Fig. 13: Computation cost for D-stream data aggregation

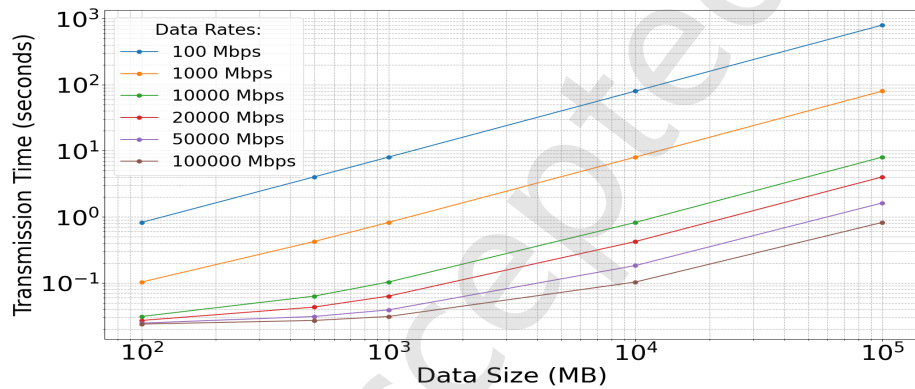


Fig. 14: Data transmission time for 10 satellites

Figure 13 illustrates the computation cost for different numbers of nodes in a network of LEO satellites acting as streaming servers, based on the D-stream algorithm. The red line, representing 10 nodes, demonstrates the highest computation cost. This is expected because with fewer nodes, each node must handle a larger portion of the total data, increasing the computational load per node. As time progresses, the computation cost decreases, reflecting the periodic clustering and density decay functions of the D-stream algorithm, which gradually reduce the computational demands. The green line, representing 50 nodes, shows a lower computation cost compared to the 10-node scenario. With more nodes, the data distribution is more efficient, reducing the computational burden on each individual node. The curve exhibits a similar downward trend over time due to the same clustering and decay mechanisms. The blue line, representing 100 nodes, exhibits the lowest computation cost among the three scenarios. This further supports the observation that increasing the number of nodes in the network significantly alleviates the computational load on each node. The initial computation cost is already lower, and the downward trend continues as the algorithm progresses. Overall, the results

demonstrate the advantages of increasing the number of nodes acting as streaming servers. More nodes lead to more efficient data handling and lower computation costs per node. The D-stream algorithm effectively manages the computational load by periodically clustering data points and applying density decay, further optimizing the network's performance over time.

6.4 Satellites Data Transmission

Efficient data transmission is a fundamental challenge in satellite communication networks, where factors such as satellite count, bandwidth, and data size significantly impact performance. Understanding how these variables interact can help optimize network efficiency and ensure reliable data transfer. In this section, we investigate the impact of data size and satellite count on the total transmission time required for data transfer in the satellite network. The results are presented in three configuration scenarios: a shard with 10, 50, and 100 satellites. The data sizes tested range from approximately 10^2 to 10^5 MB, covering typical large-scale data transmissions in satellite applications. This is significant for understanding the network's scalability in handling varying data loads.

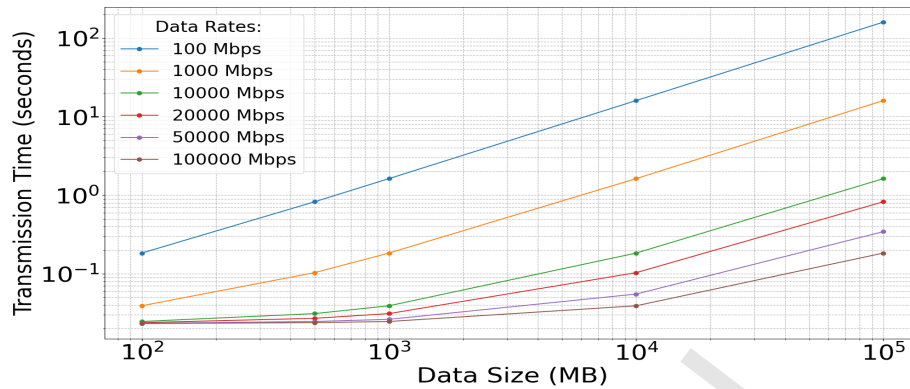


Fig. 15: Data transmission time for 50 satellites

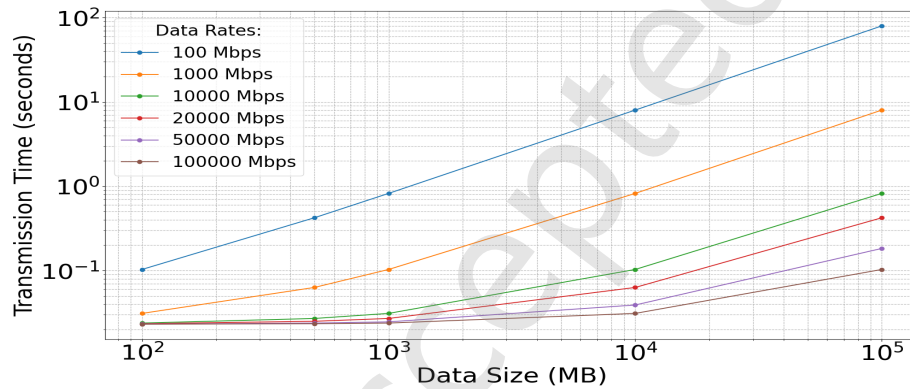


Fig. 16: Data transmission time for 100 satellites

In a 10 satellites configuration in a shard as illustrated in figure 14, transmission time increases with data size, particularly at lower bandwidths, such as 100 Mbps, where capacity limitations become evident. Higher bandwidths, like 100,000 Mbps, significantly reduce transmission delays, highlighting the critical role of bandwidth in improving efficiency. Within 50 and 100 satellites configuration (figures 15 and 16), transmission times decrease significantly compared to the 10 satellites setup. Among the refereed configurations, the 100 satellites configuration achieving the most efficient performance. The enhancement in efficiency can be attributed to the increased number of satellites, which allows for a more distributed workload and reduces the time taken to transmit data packets. This effect is particularly pronounced at higher bandwidths, where the differences in transmission times among various data sizes become more distinct. The reduction in transmission time for larger data sizes highlights the synergistic effect of combining more satellites with higher bandwidth, thereby optimizing the communication link between the satellite network and ground stations.

An examination from figures 14, 15, 16 and table 4 clearly illustrates the intricate interplay between satellite count, data

size, and bandwidth in determining data transmission efficiency in satellite communication systems. The results emphasize the need for strategic planning in satellite deployment and bandwidth allocation to optimize the performance of satellite communication networks. The implementation of blockchain technology enables flexible and dynamic resource allocation, resulting in notably reduced transmission times across large satellite networks. This flexibility is further enhanced by smart contracts that automatically regulate orbital networks. Through these contracts, satellite coverage dynamically adjusts to match real-time demand patterns, ensuring service availability for business-critical needs.

6.5 Network Performance Comparison with Existing Works

As we can see in figures 17, 18 and 19, our system exhibits superior transaction throughput, latency, and scalability compared to existing systems such as [5, 6, 8] and [10]. We chose these systems for comparison due to their similarity in employing blockchain technology, particularly focusing on scalability enhancements through sharding techniques and decentralized architectures. [5, 6] and [8], share a common

Table 4: Performance analysis of 10GB (10^4 MB == 80,000 Mb) data transmission in multi-satellite networks with different data rates

Data Rate	Number of Satellites			
	1	10	50	100
100 Mbps	819.20s	81.92s	16.38s	8.19s
1000 Mbps	81.92s	8.19s	1.64s	0.82s
10,000 Mbps	8.19s	0.82s	0.16s	0.08s
20,000 Mbps	4.10s	0.41s	0.08s	0.04s
50,000 Mbps	1.64s	0.16s	0.03s	0.016s
100,000 Mbps	0.82s	0.08s	0.016s	0.008s
Data Distribution	80,000 Mb	8,000 Mb	1,600 Mb	800 Mb

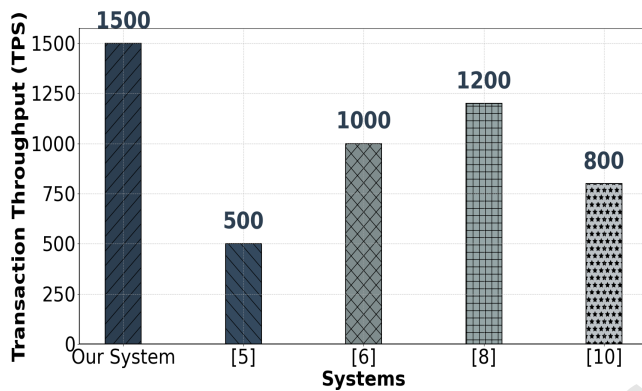


Fig. 17: Transaction throughput comparison

goal of improving the scalability of the blockchain while maintaining security and decentralization. Also as mentioned in the section 2, currently there is a scarcity of research on satellite-based blockchain networks, as this remains an emerging and innovative topic. The integration of blockchain technology with satellite communication systems is still in its early stages, with only a few studies exploring its potential applications, technical challenges, and security implications. As interest in decentralized and resilient communication infrastructures grows, further research is needed to address key issues such as network scalability, latency, and interoperability in space-based blockchain implementations.

The high transaction throughput (TPS) in our system is primarily attributed to the efficient peer-to-peer communication facilitated by the blockchain's inherent properties and the strategic geographic distribution of shards. By employing LEO satellites as streaming servers, our system ensures that data streams are effectively managed and delivered, leading to significant improvements in throughput. This decentralized approach allows for the parallel processing of transactions across multiple shards, which enhances overall system performance. Consequently, our system achieves a higher TPS compared to [5] and [8], which are restricted by their sharding techniques and consensus mechanisms that are not optimized for such large-scale distributed operations.

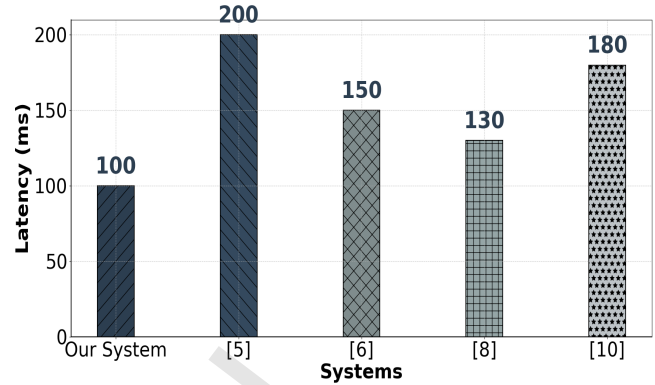


Fig. 18: Latency comparison

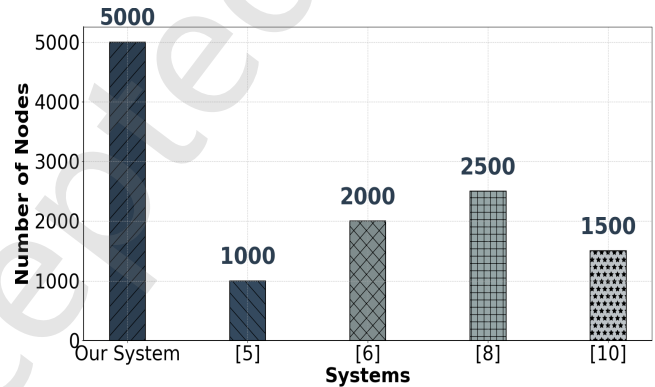


Fig. 19: Scalability comparison

Our system also demonstrates lower latency in transaction confirmation compared to [5, 6, 8] and [10]. This improvement is achieved through the reduction of resource-intensive consensus mechanisms and the efficient use of satellite-based communication. By minimizing dependence on ground stations and leveraging direct satellite-to-client communication, our system significantly reduces the time required for data transmission and processing. The use of PBFT in our inter-shard consensus mechanism further streamlines the validation process, ensuring rapid transaction finalization. These optimizations result in lower latency, enhancing the user experience and operational efficiency of the network.

OrbitalChain's ability to scale effectively with an increasing number of nodes is another standout feature that we achieved. The satellite-based architecture facilitates the distribution and management of shards across a vast geographic area, enabling the network to accommodate a larger number of nodes [44] without compromising performance. Our findings demonstrate that our proposed system is well-positioned to support the growing demands of distributed and space-based blockchain networks, paving the way for further advancements in decentralized, resilient communication infrastructures.

The table 5 presents a comparative analysis of [5, 6, 8] and

Table 5: Comparative overview of key features across different blockchain-based systems and IoT

Features	[5]	[6]	[8]	[10]	Ours
Decentralization	Yes	Yes	Yes	Yes	Yes
Scalability	Yes	Yes	Yes	Yes	Yes
Support Big Data	No	No	No	Yes	Yes
Sharding	Yes	Yes	Yes	No	Yes
Consensus Mechanism	Yes	Yes	Yes	No	Yes
Security Features	Moderate	High	High	Low	High
Private network	Yes	Yes	Yes	No	Yes
Low latency	No	No	No	Yes	Yes

[10] systems, regarding their critical features with ours. The characteristics of each system are evaluated to highlight their strengths and weaknesses. Providing insights into their suitability for different applications ranging from high-security environments to scenarios requiring efficient data handling and low latency. Our scheme's superiority over existing protocols lies in its integration of satellite technology for enhanced global coverage and reduced latency. Furthermore, the use of truth discovery scores ensures that only trustworthy satellites are prioritized, mitigating the risks associated with malicious nodes. Unlike [5] and [8], our system does not rely on complex inter-shard communication, reducing overhead and latency. Compared to the secure sharding protocol proposed by [6], our approach minimizes the reliance on resource-intensive consensus mechanisms, making it more suitable for real-time applications.

7 Future Work

In our next work, we will focus on enhancing network scalability by developing advanced and adaptable satellite trajectory optimization algorithms, which dynamically prioritize high-demand regions, while refining service responsiveness across varying conditions. Furthermore, we will introduce a series of smart contract theories designed to facilitate the seamless connection, negotiation, and organization of satellite services with clients. In addition, we plan to conduct comprehensive evaluations of OrbitalChain under several shard network conditions and stress tests to assess its robustness in real-world scenarios. Includes analyzing consensus scalability, block propagation delays, and transaction throughput.

8 Conclusion

By synergizing satellite technology with blockchain, we have developed a robust, reliable, and efficient platform. Leveraging the capabilities of LEO satellite networks, OrbitalChain offers extensive coverage and low-latency data transmission, particularly beneficial for remote and underserved regions.

The integration of innovative features, including a blockchain-based truth discovery mechanism and advanced cryptographic techniques underscores our commitment to ensuring data integrity and privacy. Experimental results demonstrate that OrbitalChain significantly improves transaction throughput, latency, and scalability, affirming its effectiveness as a robust solution for secure, high-performance multi-source data fusion in satellite-based communication networks. This work contributes to advancing the field of information fusion by offering a secure, adaptive, and efficient platform for real-time decision-making in critical applications. Overall, our work on OrbitalChain presents a compelling solution for a wide range of applications requiring secure and efficient data transmission in challenging environments.

Data availability

To ensure full reproducibility, the source code implementation is publicly available at: <https://anonymous.4open.science/r/OrbitalChain-881F/>

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. U22B2029), the Open Research Fund of The State Key Laboratory of Blockchain and Data Security, Zhejiang University.

- [1] N. A. Ochuba, D. O. Olutimehin, O. G. Odunaiya, O. T. Soyombo *et al.*, Reviewing the application of big data analytics in satellite network management to optimize performance and enhance reliability, with implications for future technology developments, *Magna Scientia Advanced Research and Reviews*, vol. 10, no. 2, pp. 111–119, 2024.
- [2] N. Tariq, M. Asim, F. Al-Obeidat, M. Zubair Farooqi, T. Baker, M. Hammoudeh, and I. Ghafir, The security of big data in fog-enabled iot applications including blockchain: A survey, *Sensors*, vol. 19, no. 8, p. 1788, 2019.
- [3] J. Liu and Z. Liu, A survey on security verification of blockchain smart contracts, *IEEE Access*, vol. 7, pp. 77 894–77 904, 2019.
- [4] B. Zhang, Y. Wu, B. Zhao, J. Chanussot, D. Hong, J. Yao, and L. Gao, Progress and challenges in intelligent remote sensing satellite systems, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 1814–1822, 2022.
- [5] M. Zamani, M. Movahedi, and M. Raykova, Rapidchain: Scaling blockchain via full sharding, in *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, 2018, pp. 931–948.
- [6] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, A secure sharding protocol for open blockchains, in

- Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 17–30.
- [7] A. Secure, The zilliqa project: A secure, scalable blockchain platform, 2018.
- [8] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, Omniledger: A secure, scale-out, decentralized ledger via sharding, in *2018 IEEE symposium on security and privacy (SP)*. IEEE, 2018, pp. 583–598.
- [9] S. Li, M. Liu, and S. Wei, A distributed authentication protocol using identity-based encryption and blockchain for leo network, in *Security, Privacy, and Anonymity in Computation, Communication, and Storage: 10th International Conference, SpaCCS 2017, Guangzhou, China, December 12-15, 2017, Proceedings 10*. Springer, 2017, pp. 446–460.
- [10] M. S. Ali, K. Dolui, and F. Antonelli, Iot data privacy via blockchains and ipfs, in *Proceedings of the seventh international conference on the internet of things*, 2017, pp. 1–7.
- [11] Chain blockchain. [Online]. Available: <https://www.chain.com/>
- [12] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, Towards scaling blockchain systems via sharding, in *Proceedings of the 2019 international conference on management of data*, 2019, pp. 123–140.
- [13] G. Xu and J. Xu, Keplerian orbits, in *Orbits: 2nd Order Singularity-free Solutions*. Springer, 2013, pp. 27–38.
- [14] D. V. Milanov, Metrics in keplerian orbits quotient spaces, *Celestial Mechanics and Dynamical Astronomy*, vol. 130, no. 3, p. 27, 2018.
- [15] M. Singhal, L. Šubr, and J. Haas, Dynamical coupling of keplerian orbits in a hierarchical four-body system: from the galactic centre to compact planetary systems, *Monthly Notices of the Royal Astronomical Society*, vol. 531, no. 1, pp. 2028–2039, 2024.
- [16] W. Wang, G. Mengali, A. A. Quarta, and J. Yuan, Analysis of relative motion in non-keplerian orbits via modified equinoctial elements, *Aerospace Science and Technology*, vol. 58, pp. 389–400, 2016.
- [17] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han, Conflicts to harmony: A framework for resolving conflicts in heterogeneous data by truth discovery, *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 1986–1999, 2016.
- [18] S. Malys, R. Wong, and S. A. True, The wgs 84 terrestrial reference frame in 2016, in *Eleventh Meeting of the International Committee on GNSS, ICG-11*, 2016, pp. 6–11.
- [19] K. M. Kelly and M. L. Dennis, Transforming between wgs84 realizations, *Journal of Surveying Engineering*, vol. 148, no. 2, p. 04021031, 2022.
- [20] 3GPP, Study on new radio (nr) to support non-terrestrial networks, *TR 38.811 V15. 2.0*, 2019.
- [21] C. Singh, C.-H. Lin, and K. Singh, Capacity and performance analysis of ris-assisted communication over rician fading channels, *arXiv preprint arXiv:2111.04783*, 2021.
- [22] M. Y. Abdelsadek, H. Yanikomeroglu, and G. K. Kurt, Future ultra-dense leo satellite networks: A cell-free massive mimo approach, in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2021, pp. 1–6.
- [23] M. Y. Abdelsadek, G. K. Kurt, and H. Yanikomeroglu, Distributed massive mimo for leo satellite networks, *IEEE Open Journal of the Communications Society*, vol. 3, pp. 2162–2177, 2022.
- [24] J. Amadid, M. Boulouird, A. Belhabib, and A. Zeroual, On channel estimation for rician fading with the phase-shift in cell-free massive mimo system, *Wireless Personal Communications*, pp. 1–21, 2022.
- [25] Y. Zhang, Y. Wu, A. Liu, X. Xia, T. Pan, and X. Liu, Deep learning-based channel prediction for leo satellite massive mimo communication system, *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1835–1839, 2021.
- [26] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, Cell-free massive mimo versus small cells, *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1834–1850, 2017.
- [27] E. Björnson, J. Hoydis, L. Sanguinetti *et al.*, Massive mimo networks: Spectral, energy, and hardware efficiency, *Foundations and Trends® in Signal Processing*, vol. 11, no. 3-4, pp. 154–655, 2017.
- [28] M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica, Discretized streams: Fault-tolerant streaming computation at scale, in *Proceedings of the twenty-fourth ACM symposium on operating systems principles*, 2013, pp. 423–438.
- [29] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, A survey on data preprocessing for data stream mining: Current status and future directions, *Neurocomputing*, vol. 239, pp. 39–57, 2017.
- [30] D. Beaver, Efficient multiparty protocols using circuit randomization, in *Advances in Cryptology—CRYPTO’91: Proceedings 11*. Springer, 1992, pp. 420–432.
- [31] H. Zhu and W. K. Ng, Beaver triple generator from multiplicatively homomorphic key management protocol, in *International Conference on Advanced Information Networking and Applications*. Springer, 2022, pp. 492–503.
- [32] P. Mohassel and Z. Secureml, A system for scalable privacy-preserving machine learning, in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 19–38.
- [33] J. Liu, M. Juuti, Y. Lu, and N. Asokan, Oblivious neural network predictions via minionn transformations, in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 619–631.
- [34] C. Orlandi, A. Piva, and M. Barni, Oblivious neural network computing via homomorphic encryption, *EURASIP Journal on Information Security*, vol. 2007, pp. 1–11, 2007.
- [35] R. Canetti, Security and composition of multiparty cryptographic protocols, *Journal of CRYPTOLOGY*, vol. 13, pp. 143–202, 2000.

- [36] C. Guan, A. Mohaisen, Z. Sun, L. Su, K. Ren, and Y. Yang, When smart tv meets crn: Privacy-preserving fine-grained spectrum access, in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 1105–1115.
- [37] Y. Lindell and B. Pinkas, A proof of security of yao's protocol for two-party computation, *Journal of cryptology*, vol. 22, pp. 161–188, 2009.
- [38] M. Castro and B. Liskov, Practical byzantine fault tolerance, *Third Symposium on Operating Systems Design and Implementation*, pp. 359–368, 1999.
- [39] X. Hao, L. Yu, L. Zhiqiang, L. Zhen, and G. Dawu, Dynamic practical byzantine fault tolerance, in *2018 IEEE conference on communications and network security (CNS)*. IEEE, 2018, pp. 1–8.
- [40] A. A. Sajo, X. Huang, M. Saif, and Q. Chen, Latency minimization in blockchain-enabled fog computing networks: A novel byzantine fault tolerance approach, *19th International Conference on Smart City*, pp. 1529–1536, 2022.
- [41] W. Zhong, C. Yang, W. Liang, J. Cai, L. Chen, J. Liao, and N. Xiong, Byzantine fault-tolerant consensus algorithms: A survey, *Electronics*, vol. 12, no. 18, p. 3801, 2023.
- [42] Y. Zheng, H. Duan, and C. Wang, Learning the truth privately and confidently: Encrypted confidence-aware truth discovery in mobile crowdsensing, *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2475–2489, 2018.
- [43] CelesTrak. (2025) Norad two-line element set archives. [Online]. Available: <https://celestrak.org/NORAD/archives/>
- [44] F. Baccelli, S. Candel, G. Perrin, and J.-L. Puget, Large satellite constellations: Challenges and impact, Ph.D. dissertation, Académie des sciences, 2024.
- [45] C. Wang, K. Ren, J. Wang, and Q. Wang, Harnessing the cloud for securely outsourcing large-scale systems of linear equations, *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1172–1181, 2012.
- [46] P. Mohassel, M. Rosulek, and Y. Zhang, Fast and secure three-party computation: The garbled circuit approach, in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 591–602.
- [47] D. Heath and V. Kolesnikov, Stacked garbling: Garbled circuit proportional to longest execution path, in *Annual International Cryptology Conference*. Springer, 2020, pp. 763–792.
- [48] H. Carter, B. Mood, P. Traynor, and K. Butler, Secure outsourced garbled circuit evaluation for mobile devices, *Journal of Computer Security*, vol. 24, no. 2, pp. 137–180, 2016.

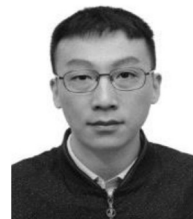
Author biography



Edson Eliezer da Silva Tavares received his B. Sc. degree in Computer Science from Hainan Tropical Ocean University in 2018, and an M. Eng. in Computer Science and Technology in UESTC, China in 2020. He is currently pursuing his PhD in Computer Science and Technology at the University of Electronic Science and Technology of China (UESTC). His research interests include Blockchain technologies, Satellite network, Network security technology, Big data security and Data privacy.



Qi Xia received her B.Sc., M.Sc., and Ph.D. degrees in Computer Science from the University of Electronic Science and Technology of China (UESTC) in 2002, 2006, and 2010, respectively. Her research interests include network security technologies, with a focus on intrusion detection and prevention systems; big data security, including data privacy and confidentiality protection; blockchain technologies, particularly smart contract security, decentralized applications (dApps), and secure transaction processing; cybersecurity in critical infrastructures such as smart grids and financial networks; and Internet of Things (IoT) security and privacy-enhancing technologies. She currently leads the National Key Research and Development Program of China in Cyber Security and has served as the principal investigator for over 30 high-impact research projects funded by the National Natural Science Foundation of China (NSFC). Dr. Xia has authored more than 40 academic papers and book chapters. She received the second prize in the National Scientific and Technological Progress Awards in 2012 and was a visiting scholar at the University of Pennsylvania (UPenn) from 2013 to 2014. She is also a regular keynote speaker at international conferences on cybersecurity and blockchain technologies.



Wei Zhang received his B.Sc. degree in Communication Engineering from Xidian University in 2009, and his M.Sc. and Ph.D. degrees in Communication Engineering from the PLA University of Science and Technology both in 2012. He is currently a Lecturer with the Department of Electronic and Optical Engineering, Space Engineering University and a member of the Key Laboratory of Intelligent Space TTC&O (Space Engineering University), Ministry of Education, Beijing, China. His research interests include signal processing in TT&C systems and space information networks.



Hu Xia received the Ph.D. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2012. He was a Visiting Scholar with the University of Minnesota, Minneapolis, MN, USA, from 2010 to 2011. He is currently an Associate Research Fellow with UESTC.



Jianbin Gao received his B.Sc., M.Sc., and Ph.D. degrees in Computer Science from the University of Electronic Science and Technology of China (UESTC) in 2005, 2008, and 2012, respectively. His research interests include blockchain technology and its applications in secure communication, network

security with an emphasis on secure data transmission protocols, machine learning with a focus on pattern recognition and anomaly detection, and distributed ledger technologies such as smart contracts and consensus mechanisms.



Bole Wilfried TIENIN completed his Bachelor's degree in Computer Engineering, at Bobo-Dioulasso Polytechnic University in 2013. He received his Master's degree in Computer Engineering at Çukurova University in 2018. He obtained his Ph.D. degree from the University of Electronic Science and

Technology of China (UESTC), majoring in Information and Communication Engineering. He is currently a postdoctoral fellow in the School of Computer Science and Engineering (cybersecurity) from UESTC. His research interests include Deep Learning and Blockchain systems.



Grace Mupoyi Ntuala obtained his B.Sc. degree in computer science from the National Pedagogic University in 2013, and the MSc in applied mathematics with an orientation on Quantum cryptography from the African Institute for Mathematical Science in Senegal in 2015. He is currently pursuing

a Ph.D. degree in Computer Science and Technology at the University of Electronic Science and Technology of China (UESTC). His research interests include blockchain scalability particularly the performance and efficiency of consensus algorithms, graph theory, network science, and the study of complex systems, fraud detection using machine learning and artificial intelligence, deep learning for network optimization and predictive analytics, as well as smart contract security, blockchain-based financial applications and Bioinformatics.



Mohammed Raouf Senouci received his B.Sc. degree in Electronics and Computer Science in 2014, and his M.Sc. degree in Telecommunication Systems and Computer Science in 2017, both from the University of Science and Technology Houari Boumediene, Algiers, Algeria. He received his Ph.D. degree in Computer Science and Technology from the University of

Electronic Science and Technology of China (UESTC) in 2024. He is currently a Postdoctoral Research Fellow with the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC). His research interests include searchable encryption, public key encryption, authenticated encryption, certificateless encryption, FPGA, chaotic systems, embedded systems, zero-knowledge proof, and federated learning.



Toussaint Elvis COMPAORE received his B.Sc. degree in Computer Science and Technology from Jiangsu University of Science and Technology in 2024. He is currently pursuing a M.Sc. degree in Computer Science and Technology at the University of Electronic Science and Technology of China (UESTC).

His research interests include blockchain technology, network security, vulnerability detection using machine learning and artificial intelligence, deep learning for network optimization and predictive analytics, as well as smart contract security and blockchain-based financial applications.