# Blockchain-based Secure Multifunctional Data Aggregation for Fog-IoT Environments

Mehdi Madjid Abbas[1] | Omar Rafik Merad Boudia[1] | Sidi Mohammed Senouci[2] | Ghalem Belalem[1]

1 LIO Laboratory, University of Oran1 Ahmed Ben Bella, Algeria

2 DRIVE Laboratory, University of Burgundy, France

Correspondence

M. M. ABBAS, LIO Laboratory, University of Oran1 Ahmed Ben Bella, Algeria.

Email: abbas.mehdi@edu.univ-oran1.dz

**Summary**

Data aggregation, in its basic form, has been widely used, and several solutions have been proposed for IoT environments. However, to calculate statistical metrics, detect anomalies, and predict future trends, we need to perform various data analysis functions on the aggregated data. Recently, multifunctional data aggregation (MFDA) has been proposed to calculate various statistical functions such as sum, mean, variance, covariance, and analyze of variance (ANOVA). The purpose of MFDA is to enable the improvement of decision making, resource allocation and system performance by providing diverse and varied statistical data. However, the existing solutions involving MFDA generate significant communication and calculation costs. Furthermore, they cannot prevent malicious aggregators from sending fake data. Recently, the Fog computing paradigm has been adopted in IoT environments to address various challenges and enhance the efficiency of data processing and storage. The blockchain technology has been integrated in various IoT applications to enhance the security, increase transparency, and facilitate decentralized data exchange and transactions. In this paper, we propose BMDA, a blockchain-based secure multifunctional data aggregation method for IoT-Fog environments. BMDA employs an encoding function to structure the data before their transmission. Furthermore, to ensure privacy preservation, authentication, data integrity and to resist malicious aggregators, we employ Paillier homomorphic encryption, BLS signature, and blockchain technology. The security analysis demonstrates the robustness of our proposal, and the performance analysis in terms of computations and communications shows the effectiveness of BMDA compared to existing solutions.

## 1. | INTRODUCTION

The advancement of interconnected and communicative objects is an ongoing process. With an increasing number of products available on the market, the development of the Internet of Things (IoT) has created new areas for investigation in information and communication sciences. In this context, a significant volume of data is generated, requiring efficient filtering, processing, and utilization [1]. The rapid growth of IoT devices in an interconnected society poses challenges that drive the scientific community and researchers to propose solutions for effectively and securely managing the generated data [2]. Data aggregation is one of the solutions being explored to efficiently handle the transmitted data from IoT devices. As depicted in Figure 1, data aggregation involves bundling transmitted messages into a single one, which improves efficiency and reduces communication costs [3].

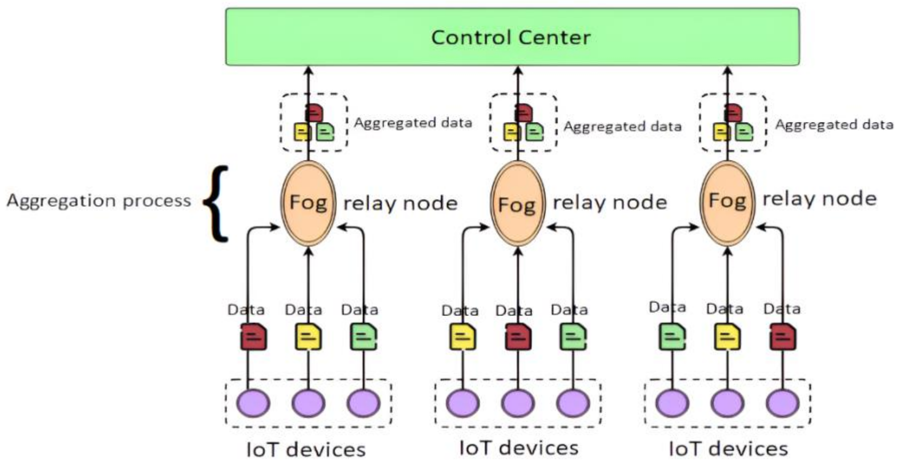


**FIGURE 1** Data aggregation process.

Fog computing has demonstrated its ability to host time-sensitive applications that require real-time data processing, highlighting the significance of secure and efficient data aggregation [4]. However, the integration of data aggregation in IoT environments, particularly within the context of Fog computing, introduces security challenges. The nature of IoT systems, characterized by inherent limitations and constraints, coupled with the distributed and interconnected nature of Fog computing, gives rise to potential vulnerabilities that must be addressed. Consequently, striking a balance between reaping the benefits of data aggregation and upholding the security of the aggregated data becomes crucial [5]. In Fog-IoT environments, real-time user/device data is collected at regular intervals, for instance, every 10 minutes. This data can encompass highly sensitive information like health records, personal preferences, credit details, and behavioral patterns. Therefore, it is crucial to protect this data from unauthorized access.

Existing data aggregation schemes in Fog-IoT environments [6]-[10], highlight the need to protect user/device data throughout the aggregation process. Many of these schemes employ homomorphic encryption to encrypt the user/device data, allowing the Fog node to aggregate the information without requiring decryption. Some solutions adopt a similar approach but incorporate blockchain technology to introduce an extra layer of security for the final aggregate storage [11]-[16]. However, these schemes only support basic aggregation operations, such as computing the summation of users/devices data. However, the control center (CC) may require more advanced statistical computations, such as calculating variance, co-variance and performing one-way ANOVA (Analysis Of Variance). The challenge lies in finding methods that allow the CC to compute advanced statistics on users/devices data without compromising the privacy of individual users, all while maintaining efficiency. This remains crucial in Fog-IoT environments. Implementing a robust security mechanism becomes a challenge due to the energy limitations of IoT devices. Many solutions [17]-[22] have been proposed to meet these issues, however, when considering the multifunctional aspect, they often incur significant communication and calculation costs. Additionally, they fail to provide protection against malicious aggregators transmitting false data.

To address the aforementioned issue, we propose BMDA, a Blockchain-based Multifunctional Data Aggregation for Fog-IoT environments. BMDA employs an encoding function to structure the data obtained from IoT devices before transmission to Fog nodes for aggregation. Encryption and signature techniques are employed to secure structured data, allowing for simplified aggregation at the Fog node level and enabling efficient and secure recovery at the CC level. The data is encrypted using the Paillier cryptosystem and structured to accommodate multiple types of data within a single report message. The report comprises a set of statistical functions, including ANOVA, applied to the raw data. To provide enhanced security, we verify the correctness of the aggregation result before storing it in a blockchain. This verification is achieved by utilizing an improved version of the Practical Byzantine Fault Tolerance (PBFT) consensus algorithm. The contributions can be summarized as follows.

- We propose a novel approach to achieve multifunctional data aggregation by combining homomorphic encryption and an encoding function. This enables the control center to efficiently recover aggregated data from the ciphertext, conserving computational and communication resources compared to previous schemes.
- We integrate blockchain technology into the network architecture to offer a strong defense against potential malicious aggregators attempting to send false data. Moreover, we enhance the traditional PBFT algorithm by incorporating a score-based technique that significantly improves the efficiency of the consensus phase, responsible

for verifying the accuracy of the aggregation result.

- We provide an in-depth theoretical analysis and experimental evaluations. The security and performance analysis demonstrate that BMDA outperforms other methods in terms of computation and communication efficiency while maintaining the security properties.

The structure of the article is organized as follows: Section 2 covers related work, Section 3 discusses models and design objectives, Section 4 presents preliminaries, Section 5 describes the proposed system, Sections 6 and 7 provide security and performance analysis, respectively, and finally, Section 8 concludes the article.

## 2. | RELATED WORK

In this section, we review previous research on secure multifunctional data aggregation and the enhancement of secure aggregation through blockchain integration within IoT-fog environments.

### 2.1. | Secure Multifunctional Data Aggregation

Recently, many solutions [17]-[22] have been proposed to calculate statistical metrics, detect anomalies, and predict future trends. These solutions provide diverse and varied statistical data to the CC. In [17], the authors propose secure one-dimensional aggregation (ODA) called MuDA, a multifunctional data aggregation scheme that addresses the security of smart grid data. Its primary objective is to allow the smart grid control center to utilize user data for calculating multiple statistical functions while ensuring user privacy. The scheme employs the BGN cryptosystem to protect sensitive data. Unlike many other secure data aggregation techniques, MuDA is designed to withstand differential attacks. Chen et al [18] introduce a secure multidimensional aggregation (MDA) approach using the Paillier cryptosystem. With this aggregation system, a utility supplier can obtain overall consumption data from all smart meters while unable to access consumption information from individual smart meters. The proposed scheme allows the smart meter to transmit various types of data in a single report message, enabling the provider to perform ANOVA analyses on the data. Privacy is ensured using the Paillier cryptosystem, multidimensional data is structured using the Super Increasing Sequence, and data integrity and authenticity are maintained through signature. In [19], the authors suggest a smart and practical Privacy-preserving Data Aggregation (PDA) scheme with a smart pricing and packaging approach for fog-based smart grids. This scheme enables diverse tariffs, multipurpose statistics, and efficiency. In the initial implementation of the smart PDA system incorporating intelligent pricing, the control center can generate more intricate and advanced aggregate data, facilitating the provision of various services, implementation of different pricing methods, and adoption of a mutually beneficial PDA scheme with Smart Pricing (PDA-SP). A feasible PDA scheme with a packing method (PDA-PM) is proposed, which reduces the quantity and size of encrypted data and enhances efficiency during secure operations. This solution employs somewhat homomorphic encryption (SHE) and a packaging technique to reduce computation and communication costs. The authors of [20] present LPM2DA, a lattice-based privacy-preserving multifunctional and multidimensional data aggregation technique. Their objective is to protect privacy while providing other security services for the smart grid, such as integrity and authentication. The proposed method allows the control center to collect spatially and temporally aggregated multidimensional data while preserving privacy. Additionally, their system enables the computation of several statistical functions, including mean, variance, and skewness, on the users' multidimensional data. Privacy is ensured using a homomorphic scheme called SHIELD (scalable homomorphic implementation of encrypted data-classifiers), based on the ring variant of the LWE (learning with error). Multidimensional data is structured using the Chinese remainder theorem, and data integrity and authenticity are guaranteed using a lattice-based digital signature from the ringLWE. In [21], Qiyu et al. suggest SEMDA (Secure and Efficient Multifunctional Data Aggregation), which is a multifunctional, and fine-grained data aggregation method integrated with lightweight cryptographic techniques. Considering the potential privacy risks associated with the close connection between IoT data, they enhance SEMDA to provide differential privacy. Their solution is based on the honest but curious model and provides confidentiality, privacy, integrity, and authentication. Privacy is ensured using the Castelluccia cryptosystem, and data integrity and authenticity are protected using MAC (Message Authentication Code). To strike a balance between data accessibility and privacy, the authors of the solution presented in [22] propose a multifunctional and multidimensional secure data aggregation system. They initially proposed a Chinese remainder theorem conversion method using a counter, which can be employed by linear homomorphic encryption algorithms to convert multidimensional data into a large integer. They then introduce a multifunctional data analysis technique that supports various aggregation functions, including continuous, linear, and polynomial functions. The authors demonstrate that the proposed method ensures privacy, integrity, authentication, and resistance against attacks, including false data injection. Privacy is preserved using the Paillier cryptosystem, multidimensional data is structured using the Chinese Remainder theorem, and data integrity and authenticity are ensured using the Schnorr signature.

## 2.2. | Blockchain-based solutions for data aggregation

The integration of edge computing and fog computing with blockchain technology in IoT environments has recently garnered significant interest to enhance efficiency and data integrity [23], [24], as well as ensure the secure aggregation of data [11]-[16]. This integration yields fortified security measures by leveraging the inherent attributes of transparency, immutability, traceability, and non-repudiation that the blockchain offers. To mitigate the negative effect of intense and sudden workload on the precision of prediction models, the authors in [11] propose an approach that combines deep learning, blockchain technology, and homomorphic encryption for data aggregation and preserves user privacy. This model is efficient in identifying smart meter manipulation and provides a low-cost, secure method for data aggregation. In [12], the authors propose EBDA, which stands for Edge Blockchain-assisted lightweight privacy-preserving Data Aggregation. The authors present a three-layer architecture that combines edge computing and blockchain technology. This architecture enables a more efficient and secure two-level data aggregation approach. The design goals are achieved by employing the Paillier cryptosystem, the one-way hash chain, and BLS signatures. The solution [13] proposes a blockchain-based novel paradigm, BNPP, for fair and secure smart grid communication. The contributions of the proposed solution include a lightweight data aggregation protocol that ensures user data privacy and communication confidentiality. An efficient non-interactive authentication mechanism is introduced, leveraging session keys for MAC authentication to ensure data integrity. The solution also incorporates a blockchain node consensus mechanism based on a subjective logic reputation model, addressing the storage of smart grid big data and mitigating single-point failure issues. The authors in [14] propose an efficient and reliable blockchain-based multidimensional data aggregation approach. Their proposal involves selecting a mining node from all smart meters to collect data using a leader election mechanism in the Raft protocol. To enable flexible dynamic user management, they employ a secret sharing homomorphism approach that can be dynamically verified. Their system supports multidimensional data aggregation, and fault tolerance, and withstands internal and external attacks. Design goals are achieved using the Threshold Verifiable Secret Sharing Homomorphism Scheme and the Schnorr signature. For a fog-enabled smart grid called DA-SADA, the authors of the solution presented in [15] propose a dual blockchain-assisted secure and anonymous data aggregation method. They establish a three-tier architecture-based data aggregation system specifically designed to enable secure and efficient data collection in smart grids, integrating blockchain technology and fog computing. The proposed mechanism ensures safety and anonymity by concurrently utilizing Paillier encryption, batch aggregation signature, and anonymous authentication, with minimal computational complexity. In [16], the authors propose a blockchain consortium-based system for smart grid data aggregation and regulation. The signcryption technique of the consortium blockchain applies to multiple receivers and multidimensional data collection. During the regulatory process, the control center, grid operator, and equipment supplier receive fixed-height blocks from the blockchain and plaintext from the encryption. Each receiver examines the multidimensional data and develops relevant control strategies for specific users. Grid administrators integrate user power regulation using smart contract feedback.

## 2.3. | Discussion

TABLE 2 compares the aforementioned works in terms of design goals, encryption methods, signature schemes, data types, use of the blockchain, and multifunctional aspects. The list of the acronyms and their definition is presented in TABLE 1. The aforementioned solutions [17]-[22] demonstrate support for the multifunctional aspect, but they exhibit performance limitations. The computation of statistical functions is dependent on individual requests for each function, leading to excessive communication and computational overhead. Additionally, these solutions do not leverage the blockchain paradigm for storing aggregates. Conversely, the solutions [12]-[16] incorporate blockchain technology to enhance aggregate security, but they lack multifunctional support and are limited to sum aggregation calculations, thereby limiting the potential utility of the aggregated data. To address these limitations, we propose a solution that combines multifunctional data processing and blockchain-based aggregate storage. We employ an encoding function to organize the data, enabling the calculation of multiple statistical functions within a single aggregation round. This approach significantly reduces both computational and communication overhead, enhancing the overall efficiency of the system.

**TABLE 1** Summary of acronyms

| Acronyms | Definition |
|---|---|
| MF | Multifunctional |
| PP | Privacy-Preserving |
| EFF | Efficiency |
| INT | Integrity |
| AUTH | Authentication |
| MA | Multiple Aggregation |
| ROB | Robustness |
| NTA | No Trust Authority |
| MFN | Multi-Function |
| FG | Fine-Grained |
| MD | Multidimensional |
| HU | High Utility |
| FT | Fault Tolerance |

| | |
|---|---|
| IA | Identity Anonymity |
| MAD | Malicious Aggregator Detection |
| CONF | Confidentiality |
| RAA | Resistance against attacks |
| Non-Re | Non-repudiation |

**TABLE 2** Comparative analysis of previous works

| Scheme | Design Goals | Encryption Method | Signature Scheme | Data type | Multifunctional | Blockchain | Advantages | Shortcomings |
|---|---|---|---|---|---|---|---|---|
| [17] | MF, PP, EFF | BGN | - | ODA | √ | - | Flexibility and Customization Security Enhancements | Communication Overhead |
| [18] | PP, INT, AUTH, EFF, MA | Paillier | Pairing-based Signature | MDA (SIS) | √ | - | Efficiency Versatility Security | Complexity Scalability Verification Process |
| [19] | PP, Diversified tariffs, MF, EFF | SHE | - | ODA | √ | - | Smart Pricing Strategies Security | Complexity Storage Overheads |
| [20] | PP, INT, AUTH, RAA | SHIELD | Lattice-based signature | MDA (CRT) | √ | - | Temporal and spatial aggregations | Complexity, Computational Costs |
| [21] | EFF, ROB, NTA, MFN, FG | Castelluccia | MAC | ODA | √ | - | Efficiency No Trusted Authority | Keys exchange complexity |
| [22] | Security, EFF, MD, MF | Paillier | Shnorr signature | MDA (CRT) | √ | - | Versatility Security | Computational Costs Scalability |
| [12] | CONF, INT, AUTH, ROB, PP, EFF | Paillier | BLS | MDA (SIS) | - | √(DVAC) | Adaptability Resistance to Attacks | Computational Costs |
| [13] | PP, INT, CONF, HU, Fairness | Modular Addition Encryption | BLS | ODA | - | √ (PBFT) | High Utility Fairness | Complexity |
| [14] | CONF, PP, AUTH, FT, NTA, RAA | Secret sharing homomorphism scheme | Shnorr signature | MDA (CRT) | - | √ Leader election algorithm in the Raft protocol | Security Dynamic User Management Fault Tolerance | Complexity Scalability Resource Requirements |
| [15] | PP, AUTH, INT, CONF, IA, ROB | Paillier | Batch aggregation signature | ODA | - | √ (PBFT) | Enhanced Security Decentralization | Complexity Scalability Resource Intensive |
| [16] | CONF, INT, Non-Re, Regu | HYBRID SIGNCRYPTION | | MDA | - | √(Smart Contract) | Enhanced Security Flexible architecture | Scalability Complexity |
| BMDA | PP, INT, AUTH, CONF, MAD, MF, EFF, HU | Paillier | BLS | ODA/ MDA | √ | √ (Enhanced PBFT) | Efficiency Security Performance Scalability | - |

## 3. | MODELS AND DESIGN GOALS

In this section, the system and attacker models will be introduced, along with the design objectives. The system model outlines the architectural framework of the proposed BMDA system and highlights the role of IoT devices, Fog nodes, blockchain, and the control center in achieving secure and efficient multifunctional data aggregation. Additionally, the attacker model delineates potential threats from malicious entities, including malicious aggregators and external attackers, and clarifies their objectives and capabilities. Furthermore, the design objectives articulate the fundamental principles guiding the development of the system, encompassing privacy preservation, data integrity, confidentiality, efficient aggregation, and resilience against various attack vectors. Through a comprehensive understanding of these models and objectives, we aim to establish a robust foundation for the subsequent discussion and evaluation of the proposed system.

## 3.1. | System model

While the majority of previous solutions that leverage Fog architecture employ a three-level architecture for their aggregation scheme, the proposed BMDA system takes a distinct approach by utilizing a four-level architecture. At the first level, numerous IoT devices ($IoTd_{ij}$) are present, followed by Fog nodes ($FN_j$) at the second level, blockchain (BC) at the third level, and the control center (CC) at the fourth level, as depicted in Figure 2. In this system, the CC has direct access to the aggregated data by querying the BC. Each FN is assigned the responsibility of covering a subset of IoT devices. The BC, in contrast, is responsible for storing the aggregated data and maintaining direct links to the Fog Nodes.

Each $IoTd_{ij}$ collects the data, encodes it, performs cryptographic operations (encryption and signature), and subsequently delivers a report to the corresponding $FN_j$. Upon receiving the reports, the node $FN_j$ undertakes report verification and homomorphically aggregates them to ensure confidentiality. Subsequently, at the initiation of the consensus phase during the PRE-PREPARE step, it disseminates these reports to other Fog nodes within the consensus group. This facilitates the computation of an aggregate copy, aiding in the detection of fraud related to the malicious aggregator. It is the responsibility of each $FN_j$ to initiate the consensus process (PBFT) and to add the resulting aggregate to the blockchain, while the other nodes act as consensus members participating in the leader-initiated voting mechanism. Once consensus is reached, and the new block is irrevocably added to the blockchain, the CC directly accesses the blockchain by performing decryption, thereby obtaining the result of the multifunctional aggregation. We consider a Trust Authority (TA), which is a fully trusted entity, responsible for generating the public/private keys used in the Paillier cryptosystem, as well as other secret parameters. However, once all system entities have gained access to the necessary secret parameters, the involvement of TA becomes unnecessary.

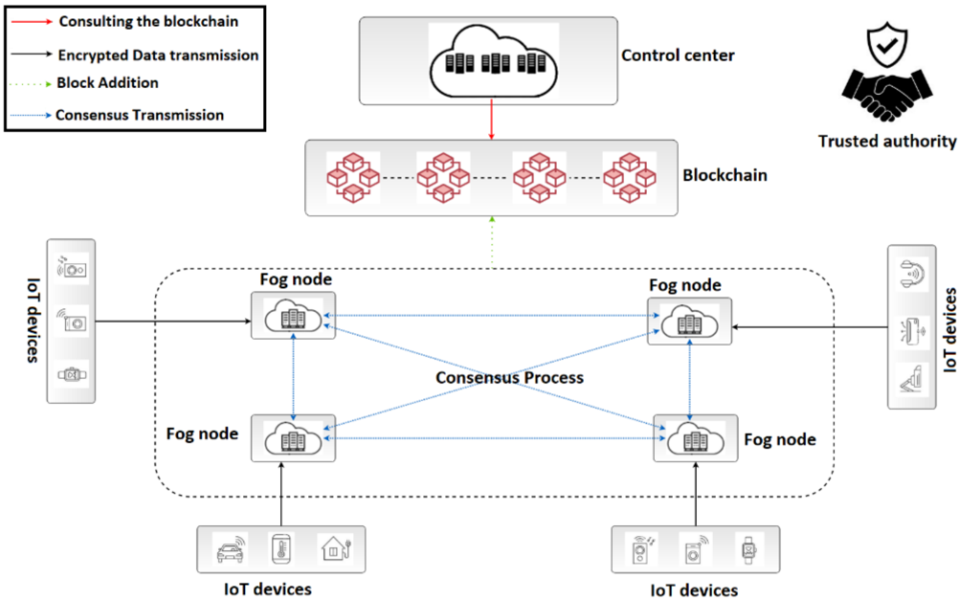


**FIGURE 2** System Model.

## 3.2. | Attacker Model

In the attacker model, we assume that FNs and CCs are honest but curious, IoT devices are honest and reliable, and external attackers are motivated to breach networks and access confidential data. False data injection attacks can manifest through the falsification or interception of user data during communication. The security considerations are as follows.

- FNs (Fog Nodes) and CCs (Control Center): These entities are assumed to be honest but curious, meaning they faithfully adhere to the protocol while exhibiting an interest in the data provided in the reports.
- A malicious aggregator node can actively participate in the data aggregation process to cause harm or act

maliciously. Its primary objective is to alter, falsify, or manipulate the aggregated data, thereby compromising the integrity and reliability of the aggregation results.

- IoT devices: These devices are considered honest and trustworthy, as they do not generate or transmit false data intentionally.
- External attackers: The existence of external attackers is acknowledged, and their objective is to infiltrate networks and gain unauthorized access to confidential data. They can eavesdrop on communication streams and attempt to identify the content of the reports.
- False data injection attacks: Involve the insertion of fabricated or manipulated data into the system, aiming to compromise the integrity and reliability of the information being processed.

### 3.3. | Design Goal

The design goal is to achieve secure data aggregation for the Internet of Things, ensuring that the CC can obtain real-time multifunctional data from a single aggregated source. To achieve this objective, the following key requirements must be met.

- Privacy-preserving: It is essential to maintain the privacy of individual data from IoT devices, even in the face of potential attacks. The CC should only have access to the aggregated plaintext stored in the blockchain, without direct access to user-specific information. Consequently, privacy breaches must be prevented.
- Integrity and Authentication: Ensuring data integrity involves detecting any anomalies or alterations that might occur during data transmission. Authentication, in contrast, focuses on validating the source of the data. This enables the Fog nodes and the CC to identify any modifications made to the reports.
- Malleability: End-to-end integrity ensures that data remains unaltered and reliable from its creation to its final use, ensuring trust in the system.
- Confidentiality: Data confidentiality must be maintained throughout the data transmission process, even when potential eavesdropping attackers are present. This necessity arises from the fact that intercepted communication channels should not grant unauthorized access to the user's confidential information.
- Malicious aggregator detection: The system should be capable of identifying a malicious aggregator that tries to insert false aggregation results into the communication.
- Multifunctional: The aggregated data should enable the CC to calculate various statistical functions, including sum, average, variance, co-variance, and ANOVA. This multifunctionality enables a more thorough statistical analysis of the data.
- Efficiency: Emphasizing the efficient utilization of resources, the proposed scheme should aim to minimize both computation and communication overhead in the IoT network. Efficiency optimization is crucial for resource optimization.
- High usefulness: The proposed system leverages the resilience and stability provided by blockchain technology, utilizing its secure data storage capabilities to effectively address the longstanding issue of single-point failure that is prevalent in traditional systems. By capitalizing on the advantages of blockchain, the system should enhance its overall usefulness and reliability.

## 4. | PRELIMINARIES

In this section, the security algorithms used in the BMDA solution are briefly presented, such as Paillier's homomorphic encryption, BLS signatures based on bilinear pairings, and the PBFT algorithm.

### 4.1. | Homomorphic encryption aggregation

Homomorphic data aggregation [25] is a processing method that allows operations to be performed on encrypted data without the need to first decrypt it. This approach ensures information confidentiality throughout the aggregation process, providing robust protection against security threats. By utilizing specially designed encryption schemes with homomorphism properties, calculations can be performed directly on the encrypted data, thereby preserving its confidentiality. This technique has applications in many areas, including smart grids, healthcare, finance, and telecommunications, where data confidentiality is of paramount importance.

### 4.2. | Paillier homomorphic cryptosystem

The Paillier Encryption Algorithm [26] is a public-key cryptography system based on asymmetric cryptography. It uses a modulo N to encrypt messages, where N is a randomly generated prime number. It is considered homomorphic because it allows operations to be performed on encrypted messages without having to decrypt them. This means that arithmetic operations can be performed on encrypted messages while maintaining data security. Here are the basic steps of the Paillier encryption algorithm:

- *Key generation*: Firstly, two random prime numbers $\boldsymbol{p}$ and $\boldsymbol{q}$ are chosen. The security of the system depends largely on the size of these prime numbers. Next, the product of these two prime numbers is calculated to obtain $\boldsymbol{n} = \boldsymbol{p} * \boldsymbol{q}$**,** and the Euler's totient function $\varphi(\boldsymbol{n})$ is also calculated, where $\varphi(\boldsymbol{n}) = (\boldsymbol{p} - 1)\ (\boldsymbol{q} - 1)$. After an integer $\boldsymbol{g}$ is chosen such that $\boldsymbol{g}$ is a generator of $\mathbb{Z}^*_{n^2}$, where $\mathbb{Z}^*_{n^2}$ is the set of natural numbers between 1 and $\boldsymbol{n^2} - 1$ that are coprime to $\boldsymbol{n^2}$. Then the value $\boldsymbol{\lambda}$ is calculated as follows: $\boldsymbol{\lambda} = \boldsymbol{lcm}\ (\boldsymbol{p} - 1, \boldsymbol{q} - 1) = (\boldsymbol{p}-1)\ (\boldsymbol{q}\ 1)\ /\ \boldsymbol{gcd}\ (\boldsymbol{p} - 1, \boldsymbol{q} - 1)$, where $\boldsymbol{lcm}$ is the least common multiple and $\boldsymbol{gcd}$ is the greatest common divisor. Afterward, $\boldsymbol{\mu}$ which is the modular inverse of the value of the function $\boldsymbol{L}(\boldsymbol{x}) = (\boldsymbol{x} - 1)\ /\ \boldsymbol{n}$, is computed modulo $\boldsymbol{n}$. In other words, $\boldsymbol{\mu}$ is the integer that satisfies the congruence $\boldsymbol{L}(\boldsymbol{\mu}) \equiv 1 \pmod{\boldsymbol{n}}$. This enables efficient decryption of encrypted messages. Finally, the public key consists of two elements: $\boldsymbol{n}$ and $\boldsymbol{g}$. The private key consists of two elements: $\lambda$ and $\boldsymbol{\mu}$.

- *Encryption:* To encrypt a message $\boldsymbol{m} \in \mathbb{Z}_n$, the transmitter calculates the encrypted message $\boldsymbol{C} = \boldsymbol{E}(\boldsymbol{m}) = \boldsymbol{g^m} * \boldsymbol{r^n}\ \boldsymbol{mod}\ \boldsymbol{n^2}$, where $r \in \mathbb{Z}^*_n$ is a random number.

- *Decryption:* To decrypt an encrypted message C, the recipient uses the private key to calculate $\boldsymbol{m} = \boldsymbol{D}(\boldsymbol{C}) = \boldsymbol{L}(\boldsymbol{c^\lambda}\ \boldsymbol{mod}\ \boldsymbol{n^2})\ \boldsymbol{\mu}\ \boldsymbol{mod}\ \boldsymbol{n}$.

## 4.3. | BLS signature scheme

Boneh-Lynn-Shacham (BLS) short signature scheme [27] is a typical bilinear pairing scheme, which uses SHA-256 hash function $H_1$: $\{0, 1\}^* \to G_1$ and $g$ is a random generator of $G_1$, and a bilinear map $e$: $G_1 \times G_1 \to G_2$. Here are the basic steps of the BLS signature algorithm:

- *Key generation:* The secret key is $x \in Z^*_q$, and compute the public key $PK = x * g$.
- *Signature generation:* The plaintext $m \in G_1$, compute the signature $\sigma = x * H(m)$.
- *Signature verification:* If $e\ (\sigma, g) = e(H(m), PK)$, then the signature is verified. Otherwise fails.

## 4.4. | Blockchain and the PBFT algorithm

The blockchain, introduced in 2008 [28], facilitates secure information exchange among nodes without relying on a central authority. It comprises a linear collection of interconnected blocks using a hash function. Consensus algorithms ensure the acceptance or rejection of new blocks. The blockchain's advantages have made it a popular technology in various IoT contexts. This paper focuses on using a hybrid blockchain that combines the transparency and immutability of a public blockchain with the confidentiality of a private blockchain, aiming to maintain data confidentiality while allowing stakeholders to verify transaction validity.

- **The Practical Byzantine Fault Tolerance (PBFT) Consensus Process**

The Practical Byzantine Fault Tolerance (PBFT) [29] consensus process is utilized. PBFT is a fault-tolerant consensus protocol designed for distributed systems in the presence of Byzantine faults. Introduced by Castro and Liskov in 1999, PBFT involves a master node proposing a block of transactions to a group of nodes. The nodes validate the proposal before reaching a consensus and appending the block to the blockchain. PBFT ensures consensus on block validity and provides resistance against malicious attacks and node failures.

The steps of the algorithm are illustrated in Figure 3 and are as follows.

- ***Client Request:*** Upon receiving a client request, the primary node broadcasts the request to all other nodes.
- ***Request Validation:*** Each node verifies the validity of the received request. If deemed valid, every node transmits a "prepare" message to all other nodes.
- ***Prepare Phase:*** Each node evaluates whether a sufficient number of nodes have issued a "prepare" message for the specific request.
- ***Commit Phase:*** Upon meeting the required threshold, each node sends a "commit" message to all other nodes.
- ***Request Execution:*** Each node verifies if a sufficient number of nodes have transmitted a "commit" message for the corresponding request. Upon satisfying the required condition, each node executes the request and communicates the outcome back to the client.

- ***Failure Handling:*** If a node fails to receive the necessary count of "prepare" or "commit" messages for a given request, it deems the request invalid.

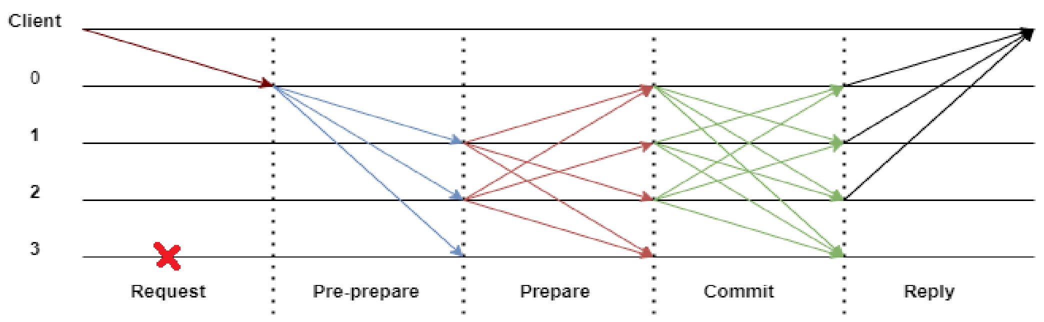


**FIGURE 3** PBFT consensus process.

## 5. | THE PROPOSED SCHEME

In this section, we present the sequence of steps of the proposed scheme, called BMDA. The scheme consists of six steps, namely System initialization, Registration, IoT device report generation, Secure data aggregation, Secure Data Storage, and finally, Data decryption and reading. TABLE 3 provides a comprehensive list of abbreviations and symbols utilized in this article, along with their corresponding definitions.

**TABLE 3** Summary of notations

| Symbol | Definition |
|---|---|
| *ODA* | One-dimensional aggregation |
| *MDA* | Multidimensional aggregation |
| *MFDA* | Multifunctional aggregation |
| *CC* | The control center |
| *FN* | The fog node |
| $CN_j$ | The PBFT consensus node |
| *BC* | The Blockchain |
| *IoTd* | The IoT device |
| $n$ | The modulus $n = p_0.q_0$ |
| $g$ | The generator of $\mathbb{Z}_{n^2}$ |
| $G_1$ | An additive group |
| $G_2$ | A multiplicative group |
| $(n,g)$ | The public key pair |
| $(\lambda, \mu)$ | The private key pair |
| $e$ | A bilinear pairing |
| $H$ | A secure hash function |
| $K$ | The number of shared keys |
| $k_1$ | The number of IoTd covered by one FN |
| $k_2$ | The number of FN covered by CC |
| $pk_{ij}$ | The public key of $IoTd_{ij}$ |
| $sk_{ij}$ | The secret key of $IoTd_{ij}$ |
| $pk_{CNj}$ | The public key of $FN_j$ for PBFT |
| $sk_{CNj}$ | The secret key of $FN_j$ for PBFT |
| $m_{ij}$ | The data of $IoTd_{ij}$ |
| $d_{ij}$ | The encoded data of $IoTd_{ij}$ |
| $S$ | The number of strategies |
| $z$ | The maximum number of bits that represent $m_{ij}$ |
| $W$ | Encoding function output (bits) |

### 5.1. | System initialization

We consider in the proposal that the trusted authority (TA) is responsible for bootstrapping the whole system. It is a fully trusted entity. It provides all system entities with the necessary secret parameters. After that, TA is not required for the aggregating process.

- *Paillier Parameter*: According to the security level, the TA selects p and q, two prime numbers, at random and distinct from one another such that gcd (p * q, (p - 1) * (q -1)) = 1, and calculate n = p * q and λ = lcm (p - 1, q-1). The TA then builds a function L(u) = u - 1/n and computes $\mu = (L(g^{\lambda} \bmod n^2)) - 1 \bmod n$ where g is the generator of $\mathbb{Z}^*_{n^2}$ such that $\gcd(L(g^{\lambda} \bmod n^2), n)=1$. The public key for encryption is (n, g), and (λ, μ) represents the private key.
- *Bilinear parameter*: TA produces the bilinear parameters ($q, P, G_1, G_2, e$) from the security parameter k by running the bilinear parameter generator Gen (k). Then, TA introduces a safe cryptographic hash function: $H: \{0, 1\}^* G_1$.
- *Blockchain parameter*: The TA initializes all Fog nodes where each node has a score of 100. It then divides them into a set of consensus nodes and a set of candidate nodes. When nodes are initialized, the order of all nodes is random.

## 5.2. | Registration

All system entities, including all IoT devices (*$IoTd_{ij}$*), Fog nodes (*$FN_j$*), and the control center (*CC*), must be registered in the *TA*.

- *Registration of CC*: The CC first chooses an $ID_{CC}$ identity. The tuple ($\lambda$, $\mu$) is then transmitted by TA.
- *Registration of $FN_j$*: The $FN_j$ initially selects an identity $ID_j$ (for all $j \in \{1,2, ..., k_2\}$). Then, *TA* determines $pk_{CNj} = sk_{CNj}P$ by selecting an integer at random $sk_{CNj} \in \mathbb{Z}^*_q$. The TA then safely sends $sk_{CNj}$ to $FN_j$
- *Registration of $IoTd_{ij}$*: The $IoTd_{ij}$ initially selects an identity $ID_{ij}$ (for all $i \in \{1,2, ..., k_1\}$). Then, *TA* determines $pk_{ij} = sk_{ij}P$ by selecting an integer at random, $sk_{ij} \in \mathbb{Z}^*_q$. The *TA* then safely sends $sk_{ij}$ to $IoTd_{ij}$. At the end of this step, the trusted authority publishes the public parameters to guarantee the proper functioning of the system: {$e, q, n, g, P, pk_{ij}, pk_{FNj}, pk_{CNj}, G_1, G_2, H$}.

## 5.3. | IoT device report generation

It is assumed that each IoT device ($IoTd_{ij}$) periodically submits its gathered data (e.g., every 15 minutes) to its corresponding Fog node. In the transmitted message, the IoT device includes both the collected data and its corresponding squared value. Notably, to ensure enhanced information security during transmission, the message undergoes encryption and digital signature calculations. The different steps are as follows.

- *Step-1:* As shown in Figure 4, representing the data encoding process, each IoT device $IoTd_{ij}$ undertakes the encoding process by adding to the predefined structure the captured data ($m_i$), its squared value ($m_i^2$), and a constant value of 1. This approach facilitates subsequent calculations and ensures the inclusion of necessary information within the encoded representation.

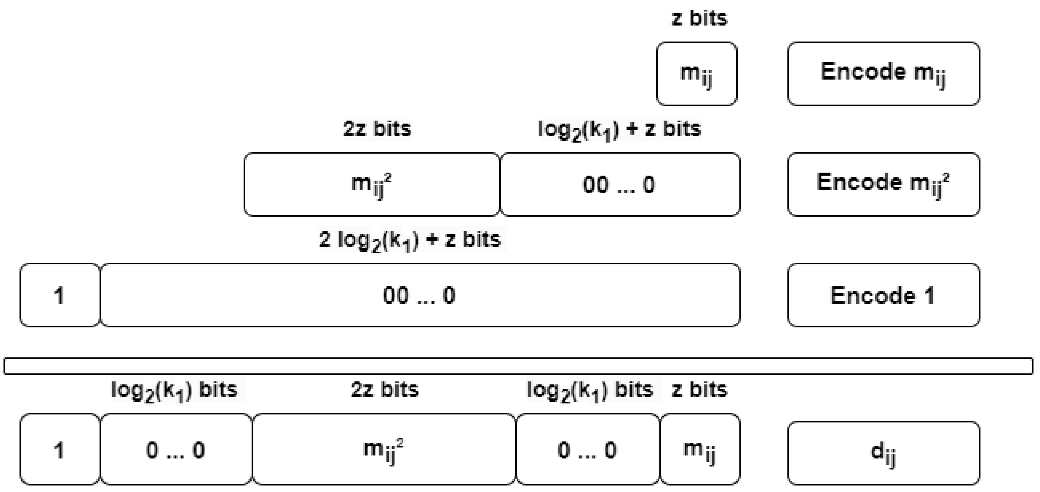


**FIGURE 4** Encoding function for BMDA.

Due to squaring, the required bits for $m_{ij}^2$ cannot be greater than 2z, where z is the maximum number of bits that can represent the data $m_{ij}$.

- *Step-2:* After randomly choosing a number $r_{ij}$ belonging to $\mathbb{Z}^*_n$, the IoT device $IoTd_{ij}$ calculates the ciphertext.

$$C_{ij} = g^{d_{ij}}.r_{ij}^{n} \, mod \, n^2 \tag{1}$$

- *Step-3:* The IoT device computes the digital signature using its private key $sk_{ij}$ as follows:

$$Sig_{ij} = sk_{ij}H(C_{ij}||ID_{ij}||TS) \tag{2}$$

- *Step-4*: Each IoT device sends its report ($Report_{ij}$) to the corresponding Fog node. The report contains the following information $\{C_{ij}, TS, ID_{ij}, Sig_{ij}\}$ where $C_{ij}$ represents the encrypted message sent, TS represents the Timestamp, $ID_{ij}$ represents the identifier of the IoT device, and finally, $Sig_{ij}$ represents the digital signature.

## 5.4. | Secure data aggregation

Following the reception of the reports, each Fog node proceeds to a batch verification to verify the validity of the signatures received, they verify the following equality:

$$e\left(P, \sum_{i=1}^{k_1} Sig_{ij}\right) = \prod_{i=1}^{k_1} e\left(pk_{ij}, H(C_{ij}||ID_{ij}||TS)\right) \tag{3}$$

It is important to note that the batch verification technique is employed to optimize calculations, reducing the number of pairings required from $2k_1$ to $k_1+1$. This reduction significantly lightens the computational overhead.

In addition to the data and their squared values, the reports received by each Fog Node (FN) also include a "1" indicator. This "1" serves the purpose of quantifying the number of IoT devices that have transmitted their reports, enabling subsequent statistical calculations to be performed.

After checking the validity, $FN_j$ aggregates all the ciphertexts. The resulting aggregate consists of three parts $\left(\sum_{i=1}^{k_1} 1 \; || \sum_{i=1}^{k_1} m_{ij}^2 || \sum_{i=1}^{k_1} m_{ij}\right)$. The following steps are executed.

- *Step-1:* The Fog node $FN_j$ aggregates the n encrypted reports as follows:

$$C_j = \prod_{i=1}^{k_1} C_{ij} mod\, n^2 \tag{4}$$

$$C_j = \prod_{i=1}^{k_1} g^{d_{ij}}.\, r_{ij}^n \, mod\, n^2$$

$$C_j = g^{\sum_{i=1}^{k_1} d_{ij} mod\, n}.\left(\prod_{i=1}^{k_1} r_{ij}\right)^n mod\, n^2$$

- *Step-2:* The Fog node $FN_j$ prepares the report containing the information $\{C_j, TS, ID_j\}$ where $C_j$ represents the encrypted aggregate, TS represents the Timestamp, and finally, $ID_j$ represents the Fog node identifier.
- *Step-3:* The Fog node securely stores the aggregate in the blockchain via the PBFT consensus process (See section 5.5).

Figure 5 illustrates an example of the resulting aggregate using the encoding function.

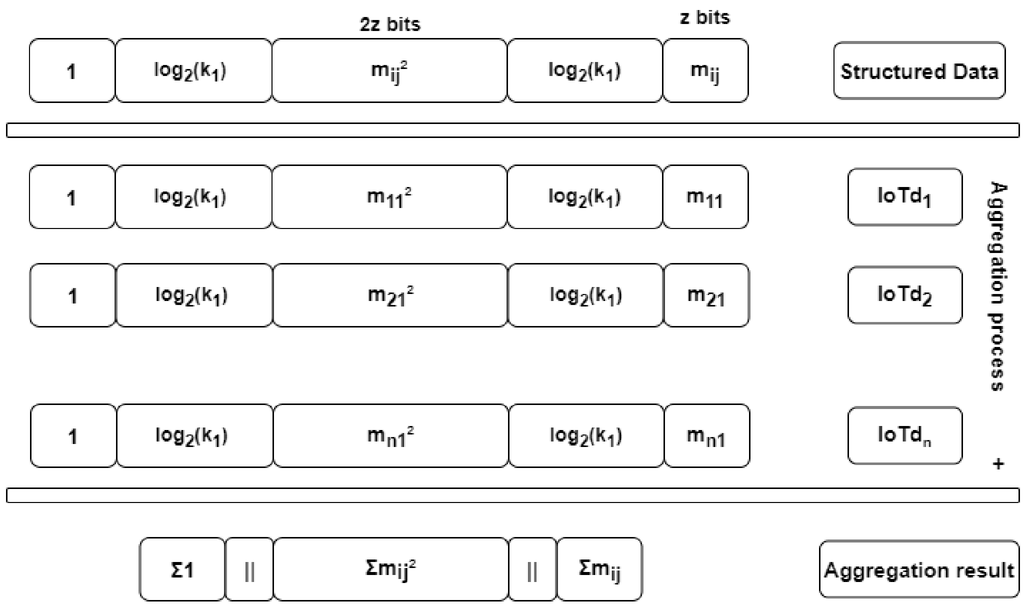


**FIGURE 5** The resulting aggregate using encoding function.

## 5.5. | Secure data storage

After computing the aggregate, every Fog node $FN_j$ utilizes the Practical Byzantine Fault Tolerance (PBFT) algorithm to securely store the aggregate in the blockchain. It is worth noting that each Fog node serves as the primary node for its respective aggregate (It directly integrates the consensus group if it is not part of it). Unlike alternative approaches that involve an additional consensus layer [14], the proposed solution designates Fog nodes to fulfill the role of consensus nodes as well. The enhanced version of the PBFT algorithm incorporates a scoring model and mainly consists of three key phases: Initialization, Block generation and verification, and Fog node reputation update.

Algorithm 1 presents the pseudocode for the Weighted Practical Byzantine Fault Tolerance (W-PBFT) consensus algorithm, while Figure 6 illustrates the main process.

*a. Initialization:*

Since the scores are predetermined by the TA, and the Fog nodes are categorized into Consensus nodes and candidate nodes, the set of consensus nodes is responsible for conducting the consensus process. In contrast, candidate nodes only update their local states upon reaching consensus, without actively participating in the consensus process. The system consists of N nodes, with a maximum allowed number of Byzantine nodes f. It is assumed that there are $k_2/2$ candidate nodes, and the number of consensus nodes must satisfy $CN \geq 3f + 1$ to ensure that the system can reach an agreement in all scenarios. Hence, we set the number of consensus nodes to be $k_2/2$.

*b. Block generation and verification:*

According to PBFT, $CN_j$ completes block generation and data verification. The active $CN_j$ participating in the consensus process represents $k_2/2$ of the number of Fog nodes in the network. The specific process is mainly divided into five steps, namely Request, Pre-prepare, Prepare, and Commit.

- *Request:* In the proposed approach, the primary node fulfills the roles of both the client and the initiator of the consensus process. In this configuration, the request step may initially appear redundant since the primary node does not need to receive a transaction request from an external client. However, even in this scenario, the first step holds significance as it allows for the assignment of a sequence number to the transaction. The primary node is responsible for assigning a unique sequence number to the transaction, which serves as its identifier throughout the subsequent stages of the consensus process. This sequence number plays a crucial role in ensuring that transactions are added to the blockchain in the order in which they were proposed. To achieve this, the primary node initiates the consensus process by sending a PRE-PREPARE message (PPM), following a series of steps. The remaining nodes in the network, acting as consensus nodes, then proceed with the prepare, and commit steps to successfully commit and add the proposed aggregate to the blockchain.

- *Pre-prepare:* Following the conclusion of the Request stage, the primary node proceeds to the pre-preparation stage. In this stage, the primary node initiates communication by transmitting PPM to the remaining consensus Fog nodes, designated as $CN_j$, representing $k_2/2$. These are the nodes actively participating in the consensus process. The PPM contains information on the block, the encrypted aggregate, and the data acquired from the IoT devices. The PPM is then sent across the network to encompass all participating consensus nodes. This action involves the following steps:

- Step-1: The primary node calculates the signature of the proposed block as follows:

$$Sig_{P0} = sk_{CN_j} H(view_{ID} || seq_{num} || E_{agg} || DATA) \tag{5}$$

Where DATA = $Report_1||Report_2||\ldots||Report_{ij}$

- Step-2: The primary node broadcasts a PRE-PREPARE message containing PPM <$view_{ID}$, $Seq_{num}$, $E_{agg}$, $Sig_{P0}$, DATA> to all consensus nodes.

• *Prepare:* Upon receiving the PPM from the primary node, each consensus node in the network embarks on the initial verification of individual BLS signatures to ensure their legitimacy, employing a batch verification process. Subsequently, mirroring the approach employed in the secure data aggregation phase, after the authentication of individual signatures, these consensus nodes aggregate the data, culminating in the acquisition of an aggregate replica, $E_{agg}{}^{r}$. This aggregate will be compared to the one proposed by the primary node. In this stage, the following steps are performed:

- Step-1: Verify the signature of the received message.

$$e(P, Sig_{P0}) = e\left(pk_{CN_j}, H(view_{ID} || seq_{num} || E_{agg} || DATA)\right) \tag{6}$$

- Step-2: Compare the aggregate proposed by the primary node ($E_{agg}$) with the aggregate calculated locally ($E_{agg}{}^{r}$) where each consensus node uses a simple formula to check their consistency. The comparison formula is expressed as follows:

$$E_{agg} \mathrel{!}= E_{agg^r} \tag{7}$$

If there are any differences between the aggregates of the consensus nodes and the primary node, it could mean that the main node is faulty or acting maliciously. Once the malicious aggregator is detected, appropriate actions can be taken to fix the issue. This involves excluding the Fog node from the aggregation process.

- Step-3: If the verification passes, the consensus node will compute the signature of the received block.

$$Sig_{CN_j} = sk_{CN_j} H(PPM) \tag{8}$$

- Step-4: Send a PREPARE message (PM) <$view_{ID}$, $Seq_{num}$, $E_{agg}$, $Sig_{CNj}$> to all other consensus nodes in the network. When a consensus node sends a PM, it also receives the prepare message broadcasted by other $CN_j$ active nodes and performs verification. It then enters the commit phase.

• *Commit:* After receiving PM messages, each $CN_j$ verifies that the number of messages is greater than 2f + 1 then sends a Commit message to the primary node. These messages indicate that the other consensus nodes have validated the block proposal and are ready to proceed to the next step. If the primary node receives more than 2f + 1 valid confirmation messages, it adds the new block to the blockchain in a secure and non-immutable way. The steps are as follows:

- Step-1: The Consensus node verifies the signature of the received messages.

$$e\left(P, Sig_{CN_j}\right) = e\left(pk_{CN_j}, H(view_{ID} || seq_{num} || E_{agg} || Sig_{P0})\right) \tag{9}$$

- Step-2: If the number of valid PMs received is greater than 2f + 1, then compute the signature of the received blocks.

$$Sig_{CN_j} = sk_{CN_j} H(PM) \tag{10}$$

- Step-3: Send a COMMIT message CM <$view_{ID}$, $Seq_{num}$, $E_{agg}$, $Sig_{CNj}$, Sig (Block||SET PM) > to the primary node in the network.
- Step-4: The primary node verifies that the number of CM received is greater than 2f + 1.
- Step-5: Add the new block to the blockchain.

The primary node verifies the authenticity of the 'commit' messages it receives. To accomplish this, the primary node ensures that the information contained within the 'commit' messages corresponds with the data initially transmitted in the 'prepare' messages. Once each consensus node effectively commits to the proposed block, and the primary node validates the 'commit' messages originating from a minimum of 2f + 1 consensus nodes, the primary node adds the block proposal into the blockchain. Through this step, the stage of the consensus process reaches its conclusion.

c. Fog node reputation update

Consensus is considered to be achieved when the primary node receives more than 2f + 1 identical confirmation messages. Subsequently, the master node disseminates the confirmation result to all nodes, including candidate nodes, and updates the scores of all nodes simultaneously. Nodes that have a confirmation result consistent with the final consensus result have their scores increased by one, while nodes with inconsistent confirmation results have their scores reduced by five. The candidate and consensus set nodes undergo updates every 100 queries. The m nodes with the lowest scores are removed from the consensus set and appended to the candidate list. Conversely, the m nodes with the highest scores in the candidate set are added to the consensus node set and assigned new numbers. At this stage, the consensus process is considered complete. The symbols utilized in the algorithm are described in TABLE 4.

**TABLE 4** Notations of the PBFT algorithm

| Symbol | Definition |
|---|---|
| PPM | Pre-prepare message |
| PM | Prepare message |
| CM | Commit message |
| $Seq_{num}$ | Sequence number |
| $E_{agg}$ | Encrypted aggregate |
| $view_{ID}$ | The current view number |
| $Sig_{P0}$ | Primary node signature |
| $Sig_{CNj}$ | Consensus node signature |
| DATA | Individual IoT device data |
| SET PM | The set of Prepare message |

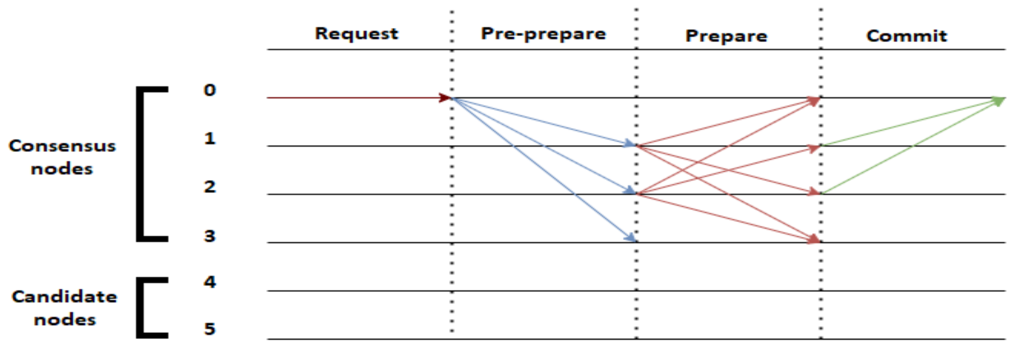


**FIGURE 6** Weighted PBFT consensus process.

It should be noted, firstly, that our approach deliberately avoids the use of traditional databases for data storage purposes. Each Fog node, a key element of the architecture, governs an exhaustive copy of the blockchain, and its role is limited exclusively to write operations when adding aggregates. As depicted in Figure 7, the detailed structure of the block, expressed by the components Header [Blocknumber, PreviousHash, Timestamp] Data [Eagg, Signagg, Signblock], constitutes the organizational basis of our data. Data aggregates, successfully undergoing the consensus process, are dutifully archived in the fog nodes, thus granting these nodes the status of highly secure storage centers for these aggregates. As for the storage configuration itself, it is essential to note that we are deviating from the use of conventional databases such as key-value databases or time series. Instead, we capitalize on the decentralized nature inherent in blockchain technology, exploiting this characteristic to distribute data fairly and securely. The approach we present is distinguished by its orientation towards decentralization, promoting the resilience and security intrinsic to the system.

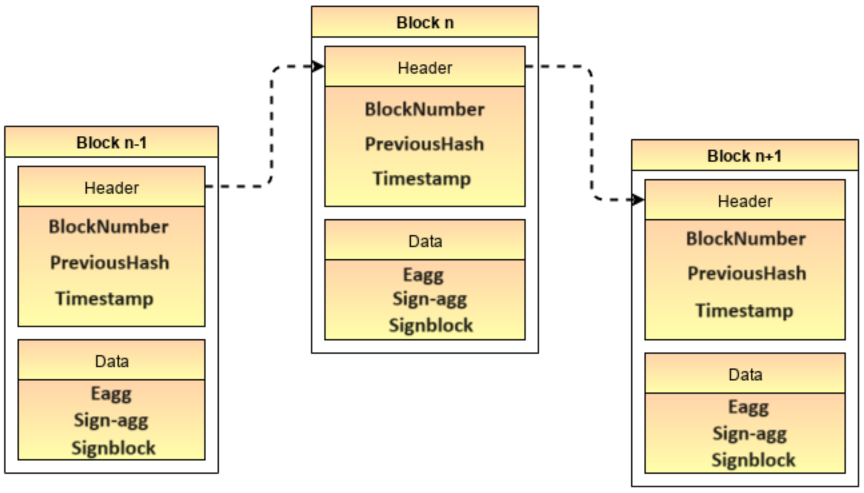


**FIGURE 7** Blockchain structure.

## 5.6. | Data decryption and reading

The CC reads information from the blockchain periodically every 15 minutes. The CC is the only one who can access it in reading mode, recover the aggregate, decipher it, and use it for multifunctional calculation. By using the corresponding public keys, the CC verifies the validity of the signatures of each block, it then accesses the blocks and retrieves the encrypted aggregate.

**Algorithm 1: Our enhanced version of the PBFT algorithm**

```
INPUT: $FN_j$, $CN_j$, scores.
OUTPUT: Updated scores, $Block_j$

INITIALIZE $FN_j$, $CN_j$, scores

nodes $\leftarrow [FN_1, FN_2, ..., FN_{k2}]$
scores $\leftarrow \{CN_1: 100, CN_2: 100, ..., CN_j: 100\}$

# Primary node
$Sig_{P0} \leftarrow sk_{CNj}H(view_{ID} \| Seq_{num} \| E_{agg} \| DATA)$
$PPM \leftarrow (view_{ID}, Seq_{num}, E_{agg}, Sig_{P0}, DATA)$
Broadcast(PPM)

# Consensus nodes
FOR j IN active_$CN_j$ DO
    $e(P, Sig_{P0}) \leftarrow e(pk_{CNj}, H(view_{ID} \| Seq_{num} \| E_{agg} \| DATA))$
    IF $E_{agg} \mathrel{!}= E_{agg}{}^r$ THEN
        $Sig_{CNj} \leftarrow sk_{CNj}H(PPM)$
        $PM \leftarrow (view_{ID}, Seq_{num}, E_{agg}, Sig_{CNj})$
        Broadcast(PM)
    END IF
END FOR

# Consensus nodes verify received PM messages and send a validation message if valid
FOR j IN active_$CN_j$ DO
    $e(P, Sig_{CNj}) \leftarrow e(pk_{CNj}, H(view_{ID} \| Seq_{num} \| E_{agg} \| Sig_{P0}))$
    IF $\sum$(PREPARE_messages) > 2f + 1 THEN
        $Sig_{CNj} \leftarrow sk_{CNj}H(PM)$
        $CM \leftarrow (view_{ID}, Seq_{num}, E_{agg}, Sig_{CNj}, Sig(Block \| SET\ PM))$
        Broadcast(CM)
    END IF
END FOR

# Primary node verifies the message and, if it receives valid COMMIT messages, updates the score
IF $\sum$(CM) > 2f + 1 THEN
    Add a new block to BC
END IF

FOR j IN active_$CN_j$ DO
    IF primary_node receives a CM from $CN_j$ THEN
```

```
        Scores[$CN_j$] ← +1
    ELSE
        Scores[$CN_j$] ← -5
    END IF
END FOR

# Every 100 requests, update the list of consensus nodes and candidate nodes
IF $Seq_{num}$ == 100 THEN
    lowest_scores ← scores sorted in ascending order and the first m are selected
    highest_scores ← scores sorted in descending order and the first m are selected
    consensus_nodes ← consensus_nodes - lowest_scores
    candidate_nodes ← candidate_nodes + lowest_scores
    consensus_nodes ← consensus_nodes + highest_scores
END IF
```

After checking the validity, CC decrypts the aggregated ciphertext $C_j$ and retrieves the aggregated data by performing the following steps:

- Step-1: The CC uses the following information to decrypt the aggregate contained in the blockchain:

$$M = \sum_{i=1}^{k_1} d_{ij} mod\ n\ and\ R = \prod_{i=1}^{k_1} r_{ij} \tag{11}$$

  The CC uses his private key the tuple ($\boldsymbol{\lambda}$, $\boldsymbol{\mu}$) to decrypt the message M. Note that the report $g^M$. $R^n$ mod $n^2$ is a ciphertext of the Paillier cryptosystem. The decryption is done as follows.

$$M = \left(C_j^{\lambda}\ mod\ n^2\right).\mu\ mod\ n \tag{12}$$

- Step-2*:* The CC retrieves both the aggregate contained in the same message, namely, $\left(\sum_{i=1}^{K_1} 1\ ||\sum_{i=1}^{k_1} m_{ij}^2 || \sum_{i=1}^{k_1} m_{ij}\right)$ using the decode function. The CC splits into 3 blocks of bits, the binary representation of $M_{agg}$ = $\left(\sum_{i=1}^{k_1} 1\ ||\sum_{i=1}^{k_1} m_{ij}^2 || \sum_{i=1}^{k_1} m_{ij}\right)$, The first block represents the aggregation of $m_{ij}$ and has a length of ($\lceil \log_2(k_1) \rceil + z$). The second block represents the aggregation of $m_{ij}^2$ and has a length of ($\lceil \log_2(k_1) \rceil + 2z$). Finally, the last block represents the aggregation of 1. The three aggregates are then recovered as follows.

$$\sum_{i=1}^{k_1} d_{ij} = \sum_{i=1}^{k_1} 1\ ||\sum_{i=1}^{k_1} m_{ij}^2 || \sum_{i=1}^{k_1} m_{ij} \tag{13}$$

- **Multifunctional calculation**

In the proposal, the control center (CC) is tasked with computing several statistical functions for analytical purposes. These functions include the sum, average, variance, covariance, and ANOVA. This section elucidates the approach adopted by the CC to facilitate these calculations.

The solution offers the advantage of providing the CC with immediate access to the information stored in the blockchain. Once the aggregates are decrypted, the CC has all the necessary data required for conducting various statistical calculations. In contrast to the approaches suggested in [17] and [19], our method eliminates the necessity for the CC to send requests to Fog nodes since it already has access to the required data.

a. *Sum Aggregation*: The Sum function is the direct consequence of the data aggregation and is calculated systematically. If the CC needs the result of the summation for any analysis, it can simply retrieve the corresponding value.

$$AGG_{sum} = \sum_{i=1}^{k_1} m_{ij} \tag{14}$$

b. *Average Aggregation*: The calculation of the average requires the value of the $AGG_{Sum}$ and the number of IoT devices having transmitted their reports. These values are recovered following the decryption of the encrypted aggregates.

$$AGG_{avg} = \frac{1}{\sum_{i=1}^{k_1} 1} . \sum_{i=1}^{k_1} m_{ij} \tag{15}$$

c. *Variance Aggregation:* If the CC needs to calculate the variance for statistical purposes, it can directly perform this calculation with the available information in $M_{agg}$. The calculation consists of applying the variance formula.

$$AGG_{var} = \frac{1}{\sum_{i=1}^{k_1} 1} . \sum_{i=1}^{k_1} m_{ij}^2 - \frac{1}{\left(\sum_{i=1}^{k_1} 1\right)^2} . \left(\sum_{i=1}^{k_1} m_i\right)^2 \tag{16}$$

d. *Coefficient-of-variation Aggregation:* The coefficient of variation is used to compare the relative spread of two data sets of different sizes, or to assess the consistency of a time series by calculating the relative stability of the mean and standard deviation. If necessary, the CC can very easily calculate the value using the data at its disposal.

$$AGG_{coeff-var} = \frac{\sqrt{\frac{1}{\sum_{i=1}^{k_1} 1} \cdot \sum_{i=1}^{k_1} m_{ij}^2 - \frac{1}{\left(\sum_{i=1}^{k_1} 1\right)^2} \cdot \left(\sum_{i=1}^{k_1} m_i\right)^2}}{\frac{1}{\sum_{i=1}^{k_1} 1} \cdot \sum_{i=1}^{k_1} m_{ij}} \tag{17}$$

e. *ANOVA Aggregation:* The calculation of ANOVA involves comparing the means of three or more groups of independent data, making it a distinct statistical measure. To initiate the calculation of ANOVA, the CC is required to send a specific request indicating its intention. By comparing the means of multiple groups of observations collected from IoT devices, the CC can evaluate whether various usage strategies are significantly influenced by the k-strategies. Within a one-way ANOVA framework, the null hypothesis posits that the means of the measured data are not significantly impacted by different usage strategies.
To perform ANOVA calculations, it is essential to compute the variance between groups.

$$SS_b = \sum_{j=1}^{s} \sum_{i=1}^{k_1} m_{ij}^2 - \frac{1}{k_1} \sum_{j=1}^{s} \left(\sum_{i=1}^{k_1} m_{ij}\right)^2 \tag{18}$$

Additionally, it is necessary to compute the variance within the group.

$$SS_w = \frac{1}{k_1} \sum_{j=1}^{s} \left(\sum_{i=1}^{k_1} m_{ij}\right)^2 - \frac{1}{k_1 . s} \left(\sum_{j=1}^{s} \sum_{i=1}^{k_1} m_{ij}\right)^2 \tag{19}$$

The (j + 1) represents the difference in usage strategy and the $m_{ij}$, $m_{ij}^2$ represents the capture data and its squared value stored in the blockchain.

The CC calculates the F-value of the F-test as $F = \frac{SS_b / (k_1 - s)}{SS_w / (s-1)}$. For the degrees of freedom s − 1 denominator and $k_1$ – s numerator, the CC rejects the null hypothesis, which means that at least one of the strategies of use has a significant influence on the users.; otherwise, the CC accepts the null hypothesis, which means that no usage strategy has a significant influence on users.

- **Multidimensional Data Aggregation**

We can expand the scope of our solution to accommodate multidimensional data. While the aggregation process and essential steps remain consistent, we must adapt the data structure accordingly. Each IoT device now measures and generates *l* types of data, denoted as ($m_{i1}$, $m_{i2}$, ..., $m_{il}$).

- Step-1: Each type of data is structured using the encoding function in (Figure 4) independently for each IoT device.

- Step-2: Let z be the maximum number of bits that could represent a data type $m_{il}$. $IoTd_{ij}$ encodes its l types of data ($m_{i1}$, $m_{i2}$, ..., $m_{il}$) into ($d_{i1}$, $d_{i2}$, ..., $d_{il}$) and constructs $D_{ij}$ as follows:

$$d_{ijk} = \left(m_{ijk}\right)_2 || 0^{\theta}, k = 1, \dots, l \tag{20}$$

$$\text{Where } \theta = (\lceil log_2(k_1) \rceil + W) * (k-1)$$

$$D_{ij} = d_{ij1} + d_{ij2} + \cdots + d_{ijl} \tag{21}$$

- Step-3: CC uses the decoding function to retrieve each aggregated data $\sum_{i=1}^{k_1} d_{ijk}$. CC divides the binary representation of $\sum_{i=1}^{k1} D_{ij}$ into l blocks of bits, the length of each block is $(\lceil log_2(k_1) \rceil + W)$. Thus, the first $(\lceil log_2(k_1) \rceil + W)$ least significant bits correspond to the aggregation result $\sum_{i=1}^{k_1} m_{ij1}$ and so on. The CC then retrieves the aggregation result of each type of data as

$$\left(\sum_{i=1}^{k_1} D_{ij}\right) = \left(\sum_{i=1}^{k_1} 1 \,||\, \sum_{i=1}^{k_1} m_{ijl}^2 \,||\, \sum_{i=1}^{k_1} m_{ijl}\right) || \dots \dots \left(\sum_{i=1}^{k_1} 1 \,||\, \sum_{i=1}^{k_1} m_{ij2}^2 \,||\, \sum_{i=1}^{k_1} m_{ij2}\right) || \left(\sum_{i=1}^{k_1} 1 \,||\, \sum_{i=1}^{k_1} m_{ij1}^2 \,||\, \sum_{i=1}^{k_1} m_{ij1}\right) \tag{22}$$

## 6 | SECURITY ANALYSIS

In this section, we conduct a comprehensive analysis of the various security aspects associated with the BMDA proposal. The objective is to demonstrate the successful attainment of the design goals concerning security, which include preserving privacy, ensuring integrity and authentication, and maintaining confidentiality.

### *a. Privacy-preserved individual IoT data*

In the BMDA proposal, we prioritize the preservation of individual IoT device data privacy, which is achieved through the utilization of Paillier homomorphic encryption. By employing this encryption technique, the data remains secure against various forms of ciphertext analysis, as the Paillier cryptosystem offers semantic security. It is important to note that no entity within the network, except for the IoT devices themselves, can access the individual data. Consequently, the captured data is encrypted before transmission to the Fog node for aggregation. The fog nodes aggregate them without the need for decryption and then, depending on how our aggregation protocol works, these aggregates are stored in the blockchain. For reading and decrypting the data, the CC will only have access to the aggregated data. Note that Paillier's cryptosystem is considered safe and reliable due to the mathematical difficulty of solving some of the problems underlying its operation, notably the difficulty of the prime number factorization problem, which is considered an NP-complete problem and difficult to solve efficiently for large primes but also the assumption that the quadratic residually problem is difficult to solve. Finally, due to the randomization of $r$ in the calculation of the Paillier ciphertext, dictionary attacks are ineffective because the encryption of the same data will induce different ciphertexts with a very high probability, which makes the exhaustive search for individual data very difficult.

### *b. Integrity and authenticity of data*

In the BMDA proposal, after completing the processes of data collection, encoding, and encryption, each IoT device signs the message $\{C_{ij}, TS, ID_{ij}, Sig_{ij}\}$ before transmitting it to the corresponding Fog node using the BLS signature algorithm [24]. The primary Fog node, upon receiving the reports, performs aggregation and then forwards a PPM message to the Fog nodes that belong to the consensus group. Sending the PPM message to the consensus nodes serves the purpose of enabling the calculation of an aggregate copy, which is essential for detecting malicious aggregates during the PBFT consensus process.

The Fog nodes first check the $ID_{ij}$ and the TS of each message (This first check guarantees the legitimacy and the freshness of the messages received). Subsequently, the integrity of the messages is checked using batch verification. All elements within the transmitted message are integral to the verification process, and any manipulation or alteration will result in verification failure. So, even though the attacker may try to send false client data from an illegal IoT device. The Fog node N will drop directly the data after verification fails.

If the verification is successful, each Fog node approves the received data and subsequently signs the message $\{C_i, TS, ID_i\}$. Afterward, the Fog node initiates the PBFT consensus process, which is aimed at storing the aggregates in the blockchain. Note that during this transition stage before final storage, a BLS signature is used to sign the individual blocks transmitted during the consensus process. The $Sig_{ij}$ of IoT devices is generated using the private key $sk_{ij}$, and the signature $Sig_{CNj}$ using the corresponding private key, namely $sk_{CNj}$. The private keys remain secret and an attacker cannot produce a valid message under any circumstances.

Due to the malleability problem of homomorphic encryption schemes, in the data aggregation phase, a compromised Fog node can easily modify an IoT report data by multiplying the ciphertext $C_{ij}$ by a value and trying to deceive the CC by a biased aggregation result. So, even though the attacker may try to run such an attack, it is clear that the attack does not succeed, because it will be detected in the Prepare phase of the consensus process. The other nodes that act as consensus nodes will determine all valid $C_{ij}$ by checking the corresponding BLS signature $Sig_{ij}$, thus, the attacker cannot broadcast a PPM message with a valid $S_{ij}$ for the modified $C_{ij}$. Consequently, the bogus data won't be accepted in the PBFT consensus and can't be added to the blockchain. As a result, tampering with IoT data by more than $f$ (CN=3$f$+1) Fog nodes is impossible.

### *c. Malleability*

It's a well-established fact that all homomorphic encryption schemes are inherently malleable. This malleability grants attackers the ability to tamper with ciphertexts by injecting false data without detection. For example, in our Paillier-based encryption, an attacker can increase the ciphertext **C** resulting from the encryption of **m** as **C' = E(m) + a** from **C = E(m)**. In data aggregation schemes, when $IoT_{ij}$ devices send encrypted data to the $FN_j$, an attacker attempts to modify the ciphertext to fool the CC which leads to false aggregates. Thus, homomorphic encryption is not secure against alteration of the ciphertext if the integrity of the ciphertext is not guaranteed via security mechanisms. In the proposed scheme, we leverage the PBFT consensus mechanism to store the aggregate in the blockchain, guaranteeing its validity and integrity. Consequently, any alteration of the ciphertext can be detected during the verification process. This detection occurs at multiple levels: firstly, at the Fog nodes upon receipt of reports from IoT devices, facilitated by the Boneh-Lynn-Shacham (BLS) signature scheme, and secondly, during the consensus process among Fog nodes where the same verification is performed. This comprehensive approach ensures the integrity of the data and protects against potential attacks on the ciphertext.

### *d. Confidentiality*

In BMDA, the CC must access the blockchain, retrieve the encrypted aggregate, and calculate equation (12) to obtain the aggregated user data. However, without specific information about the private key $(\lambda, \mu)$, no attacker can obtain user data.

### *e. Malicious aggregator detection*

The PFBT consensus process in BMDA is based on verification during the aggregation phase. Each Fog node receiving reports from its IoT devices forwards them to other participating Fog nodes in the consensus process. These nodes first verify the signatures (batch verification); if verification is successful, they calculate the aggregate for comparison with the aggregate transmitted by the primary node (the aggregator for this round of aggregation). Consequently, any alteration will fail in the PBFT consensus process, and the malicious aggregator will be detected.

*f. PBFT consensus and Blockchain*

The security of the blockchain architecture is crucial. Once the blockchain is attacked, it will lead to the leakage of user information, IoT device location information, and other private information. The security provided by our solution is preventive. The improved version of the PBFT algorithm ensures that all aggregates added to the blockchain are valid and accepted by all Fog nodes in the network. The consensus mechanism used is byzantine fault tolerant meaning that it can continue to function properly even in the presence of a small number of malicious nodes. The security of the blockchain-based BMDA solution is based on the following points.

- *Node Authentication and Identity:* Every node in the network is authenticated and has a unique identity, preventing spoofing attacks.
- *State Replication:* Data is replicated to all Fog nodes on the network participating in the consensus process, ensuring data is always available even if a node fails.
- *Approval by consensus:* Each aggregate must be approved by consensus before being added to the blockchain, which means that all nodes participating in the consensus process must agree on the acceptance of the aggregate. Aggregates are approved by a voting process, which makes it very difficult for a malicious node to manipulate the consensus process.
- *Byzantine Fault Tolerance:* PBFT is designed to resist "Byzantine" type attacks, i.e., attacks in which nodes can give false or malicious information. The consensus proposal uses voting mechanisms to allow nodes to detect and resolve conflicts between proposed transactions. In the event of a conflict, the nodes use majority rules to determine which block to accept and a scoring model to involve only the most trusted nodes in the consensus process.
- *Availability:* PBFT is designed to be highly available, as each node can serve as a leader for a given consensus round. Additionally, the consensus process is robust and can continue to function even in the presence of faulty or malicious nodes.

*g. Multifunctionality*

The objective of enabling multifunctional computation on initial data from IoT devices is achieved. Indeed, everything is implemented through the encoding function to structure the data concisely before encryption. The aggregation process is performed and the aggregate is stored in the blockchain in the form $\left(\sum_{i=1}^{K_1} 1 \;||\sum_{i=1}^{k_1} m_{ij}^2 || \sum_{i=1}^{k_1} m_{ij}\right)$, which provides the CC with all the necessary data to perform any statistical function supported by our solution.

## 7 | PERFORMANCE ANALYSIS

In this section, we analyze the performance of the BMDA proposal in terms of computation cost, communication overhead, and complexity.

### 7.1. | Computation overhead analysis

For the performance evaluation in terms of computational cost, we use the MIRACL library [30] to calculate the execution time of the cryptographic operations used in the BMDA and the previous solutions. We use for this analysis a computer with an Intel Core i5-8365U 1.60GHz × 8 processor and 8 GB of RAM. We consider for Paillier encryption a 1024-bit n and for pairing a base field size of 160 bits.

To enable a comprehensive analysis of our contribution, we compare our solution with previous schemes. By excluding the blockchain from the comparison when evaluating solutions that address multifunctional aggregation, and by considering the blockchain when examining solutions focused specifically on blockchain, we ensure a representative assessment of our approach. To show the effectiveness of the proposed scheme, we compare its performance with Muda [17], SEMDA [21], and the TRAD schemes. TABLE 5 represents the cost of cryptographic operations namely, Exponentiation in $\mathbb{Z}_{n^2}$ Pairing, Multiplication in $G_1$, Hash function, and Modular Addition. The computational cost of reverse mapping using Pollard's Lambda ($C_{pl}$) is shown in TABLE 6. The comparison in terms of computation cost is made at three levels, namely IoT devices, Fog nodes, and the Control Center.

**TABLE 5** Cost of cryptographic operations

| Symbol | Operation | Time (ms) |
|---|---|---|
| $C_e$ | Exponentiation in $Zn_2$ | 5.43 |
| $C_p$ | Pairing | 1.44 |
| $C_m$ | Multiplication in $G_1$ | 0.23 |
| $C_h$ | Hash function | 0.07 |
| $C_a$ | Modular Addition | 0.01 |

**TABLE 6** Cost of reverse mapping using the Pollard method

| Message size | Time |
|---|---|
| 8 bits | 28.6341 (milliseconds) |
| 13 bits | 34.3637(milliseconds) |
| 32 bits | 1087(milliseconds) |
| 56 bits | 39.56 (minutes) |

We divide the comparative study in this section into three parts: multifunctional data aggregation, ANOVA aggregation, and blockchain-based data aggregation.

a. Multifunctional data aggregation:

In our solution, the data required for calculating all statistical functions are transmitted from the outset in the same and unique $C_{ij}$. In Muda [17], the CC sends a specific request containing the aggregation function it wishes to compute. In SEMDA [21], the differentiation between the aggregation calculations of the different statistical functions is done at the level of the $TD_{ij}$ (Terminal Devices). Finally, in TRAD, we assume the same algorithms for encryption and signing as BMDA, the difference is that each IoT device generates two different reports, the first containing $m_{ij}$ and the second containing $m_{ij}^2$.

In BMDA, we employ the Paillier cryptosystem that requires two exponentiations in $\mathbb{Z}_{n^2}$, therefore, $2C_e$ are needed to calculate the ciphertext $C_{ij}$, and a multiplication in $G_1$, namely, $C_m$ is required for the calculation of the signature $Sig_{ij}$. In Muda, the $SM_{ij}$ requires three-group-based exponentiation and a multiplication group operation to calculate the cipher (note that in Muda, data integrity is not considered), thus, the cost is $3C_e + C_m$. In SEMDA, the computational cost depends greatly on the aggregation function calculated, it is mainly a hash operation and modular addition. For the calculation costs the SUM = $(k+1). C_h + k. C_a$, the AVG = $(k+2). C_h + (k+1). C_a$, and finally VAR = Co-VAR $(k+3). C_h + (k+2). C_a$. The TRAD solution employs the same algorithms to encrypt and sign the messages to be transmitted as our proposal, thus, the TRAD solution would require twice the calculation time required for our solution $2(2C_e+C_m)$. The results are depicted in Figure 8, demonstrating the substantial reduction in computational overhead achieved by our BMDA solution when compared to Muda [17], SEMDA [21], and TRAD. Muda [17] relies on the use of homomorphic encryption of BGN, while SEMDA [21] employs modular addition, which, although cost-effective, necessitates the exchange of keys among all entities involved. In contrast to other solutions that necessitate multiple rounds of aggregation, our approach utilizes an encoding function to facilitate multifunctional calculations within a single round of aggregation.

In our solution, each $FN_j$ verifies the integrity and the authenticity of the data received and then performs a secure aggregation of the data, which is done as follows $(\sum 1 \| \sum m_{ij}^2 \| \sum m_{ij})$. In Muda, SEMDA, and TRAD a different aggregation is carried out for each statistical function according to the requests of the CC, which induces additional calculation costs.

In our proposed scheme, batch verification requires $(k_1+1)\ C_p$ to verify the received data. The aggregation of this data is inexpensive and just requires multiplication that can be neglected. In Muda, we have different computational costs which depend on the aggregation function computed. For the mean, $(k_1-1)*C_m$ group multiplication is necessary and for the variance, this requires $2(k_1-1)C_m+2(k_1+1)C_p$. In SEMDA, there are different calculation costs for each of the functions, the SUM = $2C_h + (k_1-1)\ C_a$, the AVG = $4C_h + 2(k_1-1)\ C_a$, and finally VAR = Co-VAR = $6C_h + 3(k_1-1)C_a$. Finally, in TRAD, the cost would be $2(k_1+1)\ C_p + 4C_m$. The results are visually represented in Figure 9, showing that our solution lies between two alternatives. On one hand, BMDA surpasses both Muda [17] and TRAD in performance. This can be attributed to our approach, which systematically aggregates the data as a whole, whereas the other solutions calculate each statistical function independently. On the other hand, BMDA incurs a higher computational cost compared to SEMDA [21]. This can be explained by the utilization of hashing and modular addition operations in SEMDA, leading to a relatively lower computational overhead.

In BMDA, the CC first checks the block validity, access to the data and then decrypts it. The decryption, carried out by the CC, will have access to the data $(\sum 1 \| \sum m_{ij}^2 \| \sum m_{ij})$ and will be able to calculate the statistical functions if necessary. In Muda [17], the cost of calculation is dependent on the size of the plaintext. Indeed, decryption requires the use of the Pollard Lambda method, which is effective when the size of the plaintext is relatively small but becomes inefficient as the size increases. For SEMDA, the CC has a different calculation cost depending on the aggregation function received. Finally, in TRAD the CC receives for each function a different message requiring verification and separate decryption.

For BMDA, the verification of the block validity requires $2C_p$, and the decryption requires $1C_e$. In Muda, the aggregation requires $3(C_e + C_{pl})$. In SEMDA, the costs at the Cloud Server level are different, the SUM = $1C_h + (k_2-1)C_a$, the AVG = $2C_h + 2(k_2-1)C_a + C_m$, the VAR = $3C_h + (3k_2-2)C_a + 2C_m + C_e$ and finally the co-VAR = $3C_h + (3k_2-2)C_a + 2C_m + 2C_e$.

The results are depicted in Figure 10, it is shown that our proposal is efficient in terms of computational cost in comparison to the other solutions at the decryption level. The Muda scheme is ineffective if the size of the message to be decrypted is relatively large. The SEMDA is also characterized by inefficiency because the CC receives four separate reports (one for each statistical function) and subsequently performs decryption four times. This process significantly increases the computational cost involved.

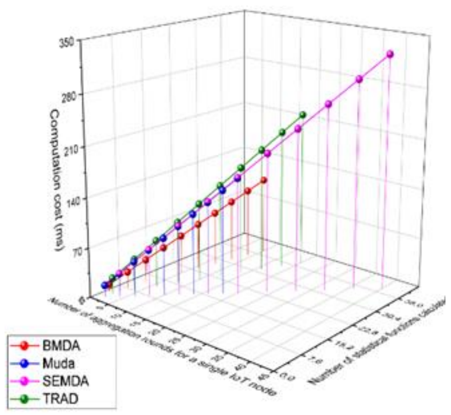


**FIGURE 8** Computation overhead at IoT device.

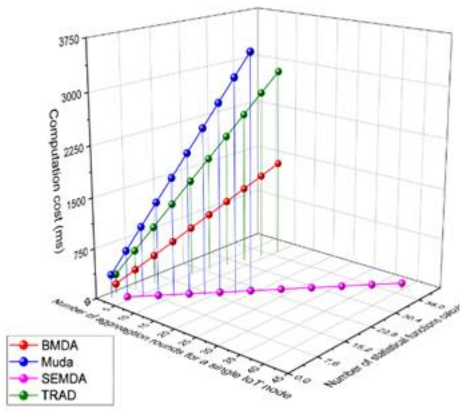


**FIGURE 9** Computation overhead at Fog nodes (k1=100).

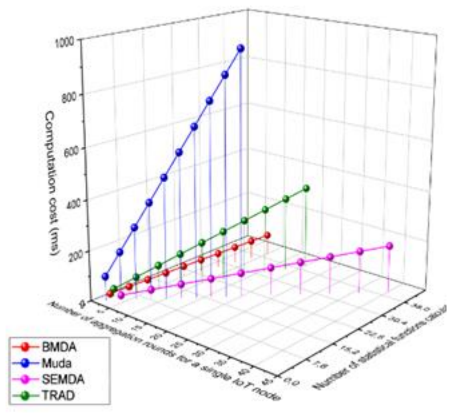


**FIGURE 10** Computation overhead at Control Center.

*b. ANOVA aggregation:*

The case of ANOVA is a special context in the sense that this function is calculated to check if a factor has a significant influence on the strategy of use. To allow the calculation of this statistical function, the control center notifies the network entities to follow a different usage strategy for a predefined period, 8 hours for example. The data $m_{ij}$ in this case represents the use under a single strategy for 8 hours. If we consider 3 different strategies, the calculation of ANOVA can be performed every 24 hours, which is the time required to gather all the essential data for the calculation.

Next, we compare our proposed ANOVA calculation approach with the Muda solution [17]. In our solution, the calculation of ANOVA is conditioned by the notification of the control center, which sends a request indicating its requirement to calculate the ANOVA. This encourages all the entities in the network to adapt accordingly. Indeed, at the IoT device level, the ANOVA aggregation is the same as that of the sum aggregation. The aggregation at the Fog node level, after receiving all the data required for ANOVA calculation, is contingent upon the adoption of k distinct strategies. This involves performing k rounds of aggregations (one for each strategy) followed by the ANOVA calculation. The total cost of the ANOVA calculation is $s((2C_e+C_m) + ((k_1+1) C_p + C_m) + (2C_p + C_e))$. In Muda [17], the control center sends a request containing the aggregation functions necessary for the ANOVA calculation to the Gateway. The total computational cost is $k_1 s C_p + (3k_1 s - s - 2) C_m$. The results are illustrated in Figure 11, depicting that the BMDA solution exhibits a lower computational cost for ANOVA computation compared to Muda [17]. This disparity can be explained by the complexity of calculations involved in Muda. It is noteworthy that our encoding function enables the acquisition of the required data for ANOVA calculations, namely $(\sum 1 || \sum m_{ij}^2 || \sum m_{ij})$, within a single round of aggregation. In contrast, Muda performs separate calculations, leading to increased computational overhead.

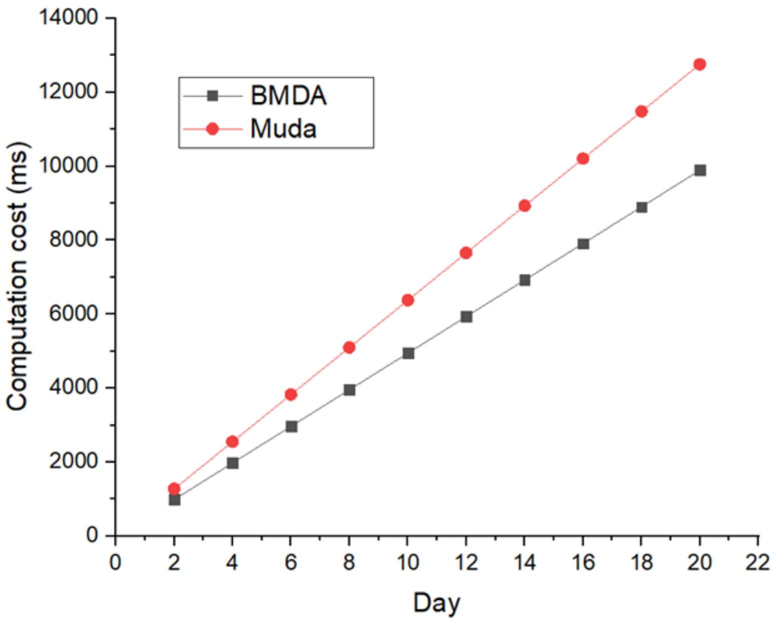


**FIGURE 11** Computation Overhead of ANOVA with (s=3, $k_1$=100).

*c. Blockchain-based data aggregation:*

In this part, we are interested in the blockchain and the PBFT consensus process. We show the effectiveness of our proposal in comparison with the traditional PBFT-based solutions [13] and [15]. The PBFT consensus process uses the BLS signature for block signing, and for verification instead of checking the signatures one by one, it uses batch verification. Moreover, since we use a score-based model, the number of nodes participating in the consensus is reduced by half. To determine the computational cost, we are only interested in the operation of the signature and verification of Fog nodes.

In the Request step, the primary node proposes a new block to add to the blockchain. For this, the node must perform a signature operation to sign the proposed block. Since each block is signed only once, the number of signing operations in this step is equal to the number of blocks offered.

In the pre-prepare stage, each consensus node that has received IoT device reports from the primary node sees its computational cost slightly increase by $(k_1+1)C_p$ due to signature verification. Aggregating this data is inexpensive and just requires multiplication which can be neglected for each round of aggregation. Also, each node receives the proposed block and must perform signature verification to validate the block. As each block is signed only once, the number of verification operations in this step is equal to the number of blocks offered. Instead of verifying each signature individually, signatures from multiple blocks can be bundled into a single packet and verified simultaneously.

In the prepare stage, each node sends a prepare message to all other nodes to indicate that the proposed block has been validated. For this, the node must perform a signature operation to sign the message. As each node sends a single prepare message, the number of signature operations in this step is equal to the number of active nodes in the network. For our solution, the cost is $(k_2/2)\ C_m$.

Similarly, in the commit step, the messages from prepare can be bundled into a single packet and checked simultaneously. This can reduce the number of signature verification operations needed at the commit stage.

Following the same operating model as the traditional PBFT, our proposal has one major difference, which lies in the number of Fog nodes participating in the consensus process. Indeed, we suggest that the number of nodes actively participating in the consensus represents half of the Fog nodes in the network. The other half is the candidate nodes. The results are shown in Figure 12, revealing that the enhanced version of the PBFT consensus process imposes a lower computational burden compared to the traditional PBFT-based solutions even in large-scale networks. This disparity can be explained by the fact that in the proposed weighted PBFT approach, the number of nodes participating in the consensus is relatively small.

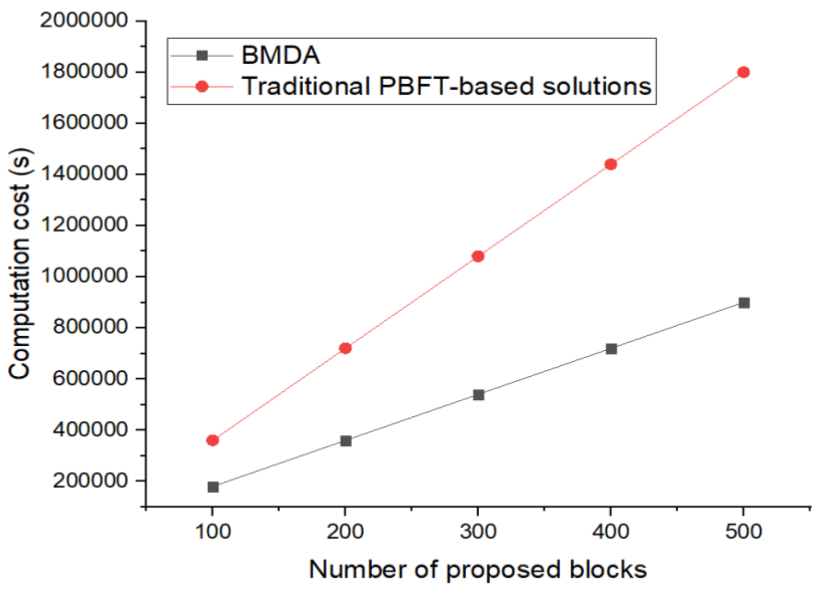


**FIGURE 12** Large-scale computation overhead of PBFT process ($k_2$=500).

## 7.2. | Communication overhead analysis

In BMDA, $m_{ij}$ and $m_{ij}^2$ are encoded, encrypted, and then transmitted periodically by each $IoTd_{ij}$ to the corresponding Fog node. We consider in the communication overhead analysis three types of communication, namely IoTd-to-FN communications, FN-to-CC communications, and FN-to-BC communications. We use the Cooja simulator [31] under Contiki2.7 (see TABLE 7) to compute the overhead at different levels. Note that we compute the overhead needed to calculate the statistical functions based on $m_{ij}$ and $m_{ij}^2$.

**TABLE 7** Cooja Simulation parameters

| PARAMETERS | VALUE |
|---|---|
| Simulator | Cooja |
| Number of nodes | IoTd: 20, 40, 60, 80, 100 |
| | FN: 2, 4, 6,…, 20, 22 |
| Simulation duration | 1 Aggregation round |
| Root node identity | 1 |
| Mote startup delay (ms) | 1 |
| Random seed | 123.456 |
| Topology | Random |
| Mote type | Skymote |
| Radio environment | UDGM |

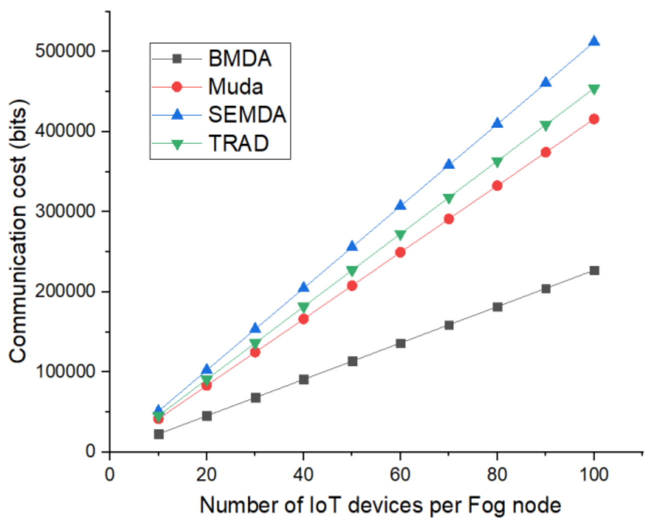
**TABLE 8** Communication overhead comparison

| Scheme | IoTd-to-FN | FN-to-BC |
|---|---|---|
| BMDA | 2272 bits | 2112 bits |
| Muda [17] | 4160 bits | 8320 bits |
| SEMDA [21] | 5120 bits | 5120 bits |
| TRAD | 4544 bits | 9088 bits |

### a. *Multifunctional data aggregation:*

In this section, we compute the overhead needed to calculate the statistical functions based on $m_{ij}$ and $m_{ij}^2$. In BMDA, the report for $m_{ij}$ and $m_{ij}^2$ contains $\{C_{ij}, Sig_{ij}, ID_{ij}, TS\}$ is transmitted from $IoTd_{ij}$ to $FN_j$ where $C_{ij} \in \mathbb{Z}_{n^2}$ and $Sig_{ij} \in G_1$. We assume that the size of $ID_{ij}$ and TS is equal to 8 bytes. So, the IoTd-to-FN communication cost is 2048+160+64 = 2272 bits. In Muda [17], the transmitted report contains only $\{C_{ij}, ID_{ij}, TS\}$. So, the communication cost is 4096+64 =4160 bits. In SEMDA [21], the transmitted message for $m_{ij}$ contains only $\{C_{ij}, \sigma_{ij}\}$. So, the communication cost is 1024+256 = 1280 bits. Note that for SEMDA an additional 1024 bits should be transmitted for key sharing since they use a symmetric encryption scheme based on modular addition. Furthermore, for $m_{ij}^2$, another cipher is calculated and then transmitted to the adjacent Fog nodes. For TRAD, we assume the same algorithms for encryption and signing as BMDA, the difference is that each IoT device sends two different reports, the first containing $m_{ij}$ and the second containing $m_{ij}^2$, so the calculation load is 2(2272) = 4544 bits. The comparison of communication costs with previous schemes and TRAD is shown in TABLE 8. The comparison in terms of communication cost IoTd-to-FN is depicted in Figure 13. The results show that BMDA incurs lower communication overhead than the other schemes. The reason is that BMDA requires only one round of aggregation to provide the CC with $m_{ij}$ and $m_{ij}^2$, while two rounds are required in SEMDA [21] and TRAD. Additionally, for the same security level, the size of the encrypted message for $m_{ij}$ and $m_{ij}^2$ is smaller with Paillier encryption compared to the BGN encryption employed in Muda [17]. As a result, the cost of BMDA is lower.

In what follows, we consider FN-to-CC communications. To make a precise comparison between BMDA and other multifunctional data aggregation schemes, namely MUDA [17] and SEMDA [21], we omit the use of blockchain and its consensus layer in BMDA. We assume that after aggregation, the Fog node directly submits the result to CC. So, each Fog node transmits, to the CC, the message containing $\{C_j, ID_j, TS\}$ where $C_j \in \mathbb{Z}_{n^2}$. We assume that the size of $ID_j$ and TS is equal to 8 bytes. So, the FN-to-CC communication cost is 2048+64 = 2112 bits. In Muda [17], the Fog node sends two aggregation reports. The first contains $\{C_{1j}, ID_j, TS\}$ for mean aggregation, and the second contains $\{C_{2j}, ID_j, TS\}$ for variance aggregation. So the communication cost is 2(4096+64) = 8320 bits. In SEMDA [21], the transmitted message contains only $\{C_j, \sigma_j\}$. The communication cost is therefore 1024+256 = 1280 bits. For each statistical function, a different report is transmitted to the CC. So, the total communication cost is 4(1280+256) = 5120 bits. Finally, in TRAD the Fog node transmits a different report for each aggregation function inducing a significant computational load. The transmitted messages have a total cost of 4(2272) = 9088 bits. Figure 14 shows the efficiency of BMDA (without the consensus layer) in terms of FN-to-CC communications cost compared to MUDA, SEMDA, and TRAD.



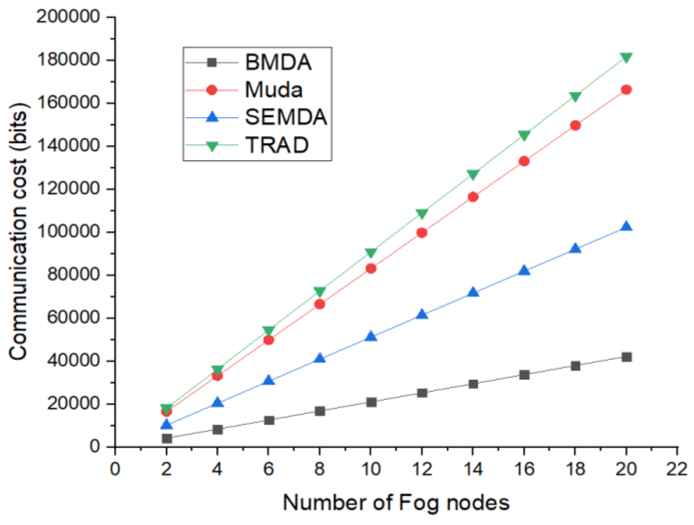


**FIGURE 13** IoTd-to-Fog communication overhead.

**FIGURE 14** Fog-to-CC communication overhead.

### b. *PBFT consensus process:*

In BMDA, we consider a blockchain with its consensus algorithm. So, in this section, we focus on the layer responsible for processing the consensus algorithm. Recall that the proposed consensus algorithm aims to verify the correctness of the aggregation result and the secure data storage on the blockchain. In Figure 15, we compare the cost of this verification with the one generated by traditional PBFT-based solutions [13] and [15]. In our solution, the number of fog nodes participating in the consensus process is less than half compared to traditional PBFT solutions. As a result, the number of transmitted messages needed to calculate the consensus is much lower in our solution. Figure 15 illustrates the increase in the number of messages as the number of fog nodes in the topology increases. We observe a moderate increase in our solution and a rapid increase in solutions based on traditional PBFT. Even in a large-scale network with a significant number of Fog nodes.

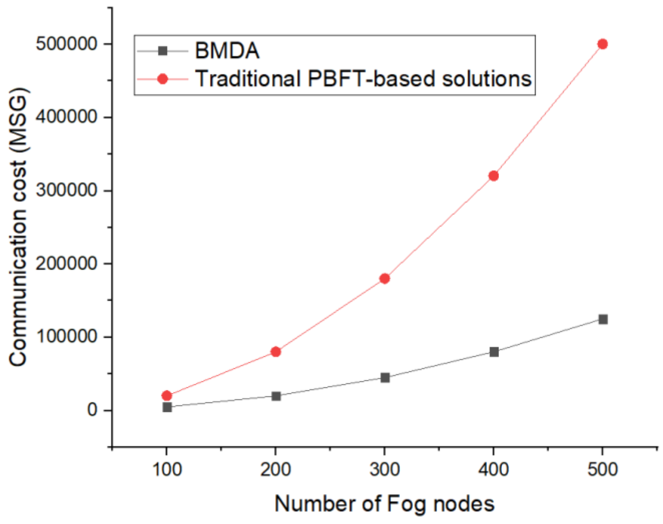


**FIGURE 15** Large-scale PBFT-process communication overhead.

Note that in [13] and [15], a malicious aggregator (e.g. who modifies an IoT report data) cannot be detected. In BMDA, an additional communication cost occurs in the pre-prepare phase of the consensus process. It is of the order of $k_1((N/2) -1))$ messages. This number of messages represents the transfer of the reports (Data) by the primary Fog node to the consensus nodes for the calculation of a copy of the aggregate, which allows the detection of malicious aggregators.

## 7.3. | Complexity analysis

This section provides an in-depth analysis of the complexities involved in various aspects of the proposed solution.

### a. *Data Encoding Complexity*

The complexity of data encoding, represented by the Encoding Function, is denoted as O $(z + \log_2(k_1))$. From an efficiency perspective, this complexity indicates linear efficiency concerning the input data size (z) and a logarithmic component $(\log_2(k_1))$ relative to the number of IoT devices $(k_1)$. This implies reasonable efficiency considering the operations involved in adding zeros and concatenating the parts. While scalability hasn't been explicitly mentioned, the logarithmic component $(\log_2(k_1))$ indicates that the encoding process can scale reasonably well with an increase in the number of IoT devices (See TABLE 9). For instance, with 250 IoT devices and z= 16bits, $l$ = 15 data types can be supported if multidimensional aggregation is considered.

**TABLE 9** Scalability analysis

| | | | EF output (bits) | |
|---|---|---|---|---|
| \|n\| | z (bits) | $k_1$ | $W$ | $l$ |
| 1024 | 16 | 250 | 65 | 15 |
| | | 500 | 67 | 15 |
| | | 750 | 69 | 14 |
| | | 1000 | 69 | 14 |
| | 32 | 250 | 113 | 9 |
| | | 500 | 115 | 8 |
| | | 750 | 117 | 8 |
| | | 1000 | 117 | 8 |

### *b. Encryption Complexity*

By employing Paillier encryption with addition-only operations, encryption complexity is $O(n^2)$, where n represents the length of the keys in bits. For decryption, complexity is $O(n^3)$, while homomorphic addition complexity is O(n). Despite the linear increase in complexity with the number of homomorphic additions, it remains dominated by the key length (n), underscoring the strength of Paillier encryption for such operations.

### *c. Signature Complexity*

In the key generation phase, the complexity typically scales with $O(\log_q)$, where q signifies the order of the multiplicative group on the elliptic curve. Generating a BLS signature involves a scalar multiplication operation between the private key and a hashed message, generally denoted as $O(\log_q)$. Verification complexity depends on the elliptic curve and may be $O(\log_{2q})$.

### *d. Blockchain Operations Complexity*

Communication between nodes and broadcasting messages (PPM, PM, CM) both have a complexity of $O(n)$, where $n$ is the total number of nodes. Message verification involves signature generation and verification, typically with a complexity of O (1) for each node upon message receipt. Message validation by consensus nodes, with the sum of PREPARE_messages, has a complexity of O(n) within the FOR loop. Score update complexity is linear (O(n)) as each consensus node is verified, adjusting scores accordingly. Adding a new block to the Blockchain generally has a complexity of O (1), dependent on the Blockchain's data structure. Updating consensus nodes and candidate nodes exhibits complexity O (m*log(m)), where m represents the number of nodes to add or remove, necessitating sorting scores.

In summary, our proposed BMDA solution significantly reduces computational overhead compared to Muda, SEMDA, and TRAD, as demonstrated in Figures 8-12. This efficiency is achieved through the use of the encoding function, which enables multifunctional calculations within a single round of aggregation. Additionally, the enhanced PBFT consensus process further reduces the computational burden, particularly in large-scale networks, by decreasing the number of nodes actively participating in consensus.

Moreover, BMDA exhibits a significantly lower communication overhead compared to Muda, SEMDA, and TRAD, in both IoTd-to-FN and FN-to-CC communications, as illustrated in Figures 13 and 14. This efficiency is attributed to BMDA's single-round aggregation for $m_{ij}$ and $m_{ij}^2$ and the smaller message size due to Paillier encryption.

Furthermore, BMDA demonstrates efficient and scalable performance across various computational aspects. The data encoding complexity of $(z+\log_2(k_1))$ ensures reasonable efficiency and scalability as the number of IoT devices increases. The encryption complexity of $(n^2)$ for Paillier encryption, combined with the complexities of signature and blockchain operations, underscores the practicality of our approach for large-scale applications. This is illustrated in TABLE 9, which shows how these efficiencies contribute to the robustness of BMDA in handling multifunctional and multidimensional aggregation and secure data transmission.

## 8 | CONCLUSION

In this research paper, we have presented a novel secure multifunctional data aggregation scheme called BMDA for Fog-IoT environments. Unlike existing schemes, BMDA does not require multiple requests of aggregation from the control center to compute statistical functions on the aggregated data, such as mean, variance, and covariance. Through the utilization of an encoding function in conjunction with the Paillier cryptosystem, all the necessary data for computation is encapsulated within a single ciphertext, enabling a single round of aggregation to be sufficient. Furthermore, blockchain technology is employed for secure data storage, and a consensus algorithm is used to verify the correctness of the aggregation result. Through theoretical analysis and simulation experiments, we have demonstrated that BMDA is efficient compared to several related works in terms of computation and communication, while still maintaining its security properties. In future research, we will focus on enhancing secure data aggregation techniques to withstand dishonest security models, employing advanced cryptographic methods, anomaly detection algorithms, and integrating machine learning for proactive threat mitigation, aiming to strengthen the resilience and trustworthiness of IoT systems.

## REFERENCES

[1] Li, S., Xu, L. D., & Zhao, S. (2015). The internet of things: a survey. *Information Systems Frontiers: A Journal of Research and Innovation*, *17*(2), 243–259. https://doi.org/10.1007/s10796-014-9492-7

[2] Tan, L., & Wang, N. (2010). Future internet: The internet of things. *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, *5*, V5-376-V5-380.

[3] Pourghebleh, B., & Navimipour, N. J. (2017). Data aggregation mechanisms in the Internet of things: A systematic review of the literature and recommendations for future research. *Journal of Network and Computer Applications*, *97*, 23–34. https://doi.org/10.1016/j.jnca.2017.08.006

[4] Stojmenovic, I., Wen, S., Huang, X., & Luan, H. (2016). An overview of Fog computing and its security issues. *Concurrency and Computation: Practice & Experience*, *28*(10), 2991–3005. https://doi.org/10.1002/cpe.3485

[5] Mohamad Noor, M. B., & Hassan, W. H. (2019). Current research on Internet of Things (IoT) security: A survey. *Computer Networks*, *148*, 283–294. https://doi.org/10.1016/j.comnet.2018.11.025

[6] Lu, R., Heung, K., Lashkari, A. H., & Ghorbani, A. A. (2017). A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT. IEEE Access: Practical Innovations, Open Solutions, 5, 3302–3312. https://doi.org/10.1109/access.2017.2677520

[7] Saleem, A., Khan, A., Malik, S. U. R., Pervaiz, H., Malik, H., Alam, M., & Jindal, A. (2020). FESDA: Fog-enabled secure data aggregation in smart grid IoT network. IEEE Internet of Things Journal, 7(7), 6132–6142. https://doi.org/10.1109/jiot.2019.2957314

[8] Guan, Z., Zhang, Y., Wu, L., Wu, J., Li, J., Ma, Y., & Hu, J. (2019). APPA: An anonymous and privacy preserving data aggregation scheme for fog-enhanced IoT. Journal of Network and Computer Applications, 125, 82–92. https://doi.org/10.1016/j.jnca.2018.09.019

[9] Zeng, Z., Liu, Y., & Chang, L. (2022). A robust and optional privacy data aggregation scheme for fog-enhanced IoT network. IEEE Systems Journal, 17(1), 1–11. https://doi.org/10.1109/jsyst.2022.3177418

[10] Abbas, M. M., Merad-Boudia, O. R., & Gasmi, S. (2022). Secure multidimensional data aggregation in IoT-fog environments using ECIES. *2022 First International Conference on Computer Communications and Intelligent Systems (I3CIS)*, 7–12.

[11] Singh, P., Masud, M., Hossain, M. S., & Kaur, A. (2021). Blockchain and homomorphic encryption-based privacy-preserving data aggregation model in smart grid. *Computers & Electrical Engineering: An International Journal*, *93*(107209), 107209. https://doi.org/10.1016/j.compeleceng.2021.107209

[12] Lu, W., Ren, Z., Xu, J., & Chen, S. (2021). Edge blockchain assisted lightweight privacy-preserving data aggregation for smart grid. *IEEE Transactions on Network and Service Management*, *18*(2), 1246–1259. https://doi.org/10.1109/tnsm.2020.3048822

[13] Bao, H., Ren, B., Li, B., & Kong, Q. (2022). BBNP: A blockchain-based novel paradigm for fair and secure smart grid communications. *IEEE Internet of Things Journal*, *9*(15), 12984–12996. https://doi.org/10.1109/jiot.2021.3107301

[14] Zhang, X., You, L., & Hu, G. (2022). An efficient and robust multidimensional data aggregation scheme for smart grid based on blockchain. *IEEE Transactions on Network and Service Management*, *19*(4), 1–1. https://doi.org/10.1109/tnsm.2022.3217312

[15] Chen, S., Yang, L., Zhao, C., Varadarajan, V., & Wang, K. (2022). Double-blockchain assisted secure and anonymous data aggregation for fog-enabled smart grid. *Engineering (Beijing, China)*, *8*, 159–169. https://doi.org/10.1016/j.eng.2020.06.018

[16] Fan, M., & Zhang, X. (2019). Consortium blockchain based data aggregation and regulation mechanism for smart grid. *IEEE Access: Practical Innovations, Open Solutions*, *7*, 35929–35940. https://doi.org/10.1109/access.2019.2905298

[17] Chen, L., Lu, R., Cao, Z., AlHarbi, K., & Lin, X. (2015). MuDA: Multifunctional data aggregation in privacy-preserving smart grid communications. *Peer-to-Peer Networking and Applications*, *8*(5), 777–792. https://doi.org/10.1007/s12083-014-0292-0

[18] Chen, Y., Martinez, J.-F., Castillejo, P., & Lopez, L. (2019). A homomorphic based multiple data aggregation scheme for smart grid. *IEEE Sensors Journal*, *19*(10), 1–1. https://doi.org/10.1109/jsen.2019.2895769

[19] Zhao, S., Li, F., Li, H., Lu, R., Ren, S., Bao, H., Lin, J.-H., & Han, S. (2021). Smart and practical privacy-preserving data aggregation for fog-based smart grids. *IEEE Transactions on Information Forensics and Security*, *16*, 521–536. https://doi.org/10.1109/tifs.2020.3014487

[20] Darzi, S., Akhbari, B., & Khodaiemehr, H. (2022). LPM2DA: a lattice-based privacy-preserving multi-functional and multi-dimensional data aggregation scheme for smart grid. *Cluster Computing*, *25*(1), 263–278. https://doi.org/10.1007/s10586-021-03387-0

[21] Wu, Q., Zhou, F., Xu, J., Wang, Q., & Feng, D. (2022). Secure and efficient multifunctional data aggregation without trusted authority in edge-enhanced IoT. *Journal of Information Security and Applications*, *69*(103270), 103270. https://doi.org/10.1016/j.jisa.2022.103270

[22] Peng, C., Luo, M., Vijayakumar, P., He, D., Said, O., & Tolba, A. (2022). Multifunctional and multidimensional secure data aggregation scheme in WSNs. *IEEE Internet of Things Journal*, *9*(4), 2657–2668. https://doi.org/10.1109/jiot.2021.3077866

[23] Heidari, A., Navimipour, N. J., Jamali, M. A. J., & Akbarpour, S. (2023). A green, secure, and deep intelligent method for dynamic IoT-edge-cloud offloading scenarios. *Sustainable Computing Informatics and Systems*, *38*(100859), 100859. https://doi.org/10.1016/j.suscom.2023.100859

[24] Gardas, B. B., Heidari, A., Navimipour, N. J., & Unal, M. (2022). A fuzzy-based method for objects selection in blockchain-enabled edge-IoT platforms using a hybrid Multi-Criteria Decision-Making model. *Applied Sciences (Basel, Switzerland)*, *12*(17), 8906. https://doi.org/10.3390/app12178906

[25] Li, F., Luo, B., & Liu, P. (2010). Secure information aggregation for smart grids using homomorphic encryption. *2010 First IEEE International Conference on Smart Grid Communications*, 327–332.

[26] Paillier, P. (2007). Public-key cryptosystems based on composite degree residuosity classes. In Advances in Cryptology — EUROCRYPT '99 (pp. 223–238). Springer Berlin Heidelberg.

[27] Boneh, D., Gentry, C., Lynn, B., & Shacham, H. (2003). Aggregate and verifiably encrypted signatures from bilinear maps. In Lecture Notes in Computer Science (pp. 416–432). Springer Berlin Heidelberg.

[28] Nakamoto, S., & bitcoin.org, W. (n.d.). *Bitcoin: A peer-to-peer electronic cash system*. Pubpub.org. Retrieved July 6, 2023, from https://assets.pubpub.org/d8wct41f/31611263538139.pdf

[29] Castro, M., & Liskov, B. (1999, February). Practical byzantine fault tolerance. In OsDI (Vol. 99, No. 1999, pp. 173-186).

[30] MIRACL: MIRACL Cryptographic SDK: Multiprecision Integer and Rational Arithmetic Cryptographic Library is a C software library that is widely regarded by developers as the gold standard open source SDK for elliptic curve cryptography (ECC). (n.d.).

[31] Contiki-NG. (n.d.). Contiki-ng.org. Retrieved June 25, 2023, from https://www.contiki-ng.org/