Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 14, No. 4, 2026, 633-650.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

BCOFF: A Blockchain-Based Framework with Consensus Protocol to Enhance Efficiency and Ensure Integrity in Fog Computing Offloading

Sajjad Daliri and Somayyeh Jafarali Jassbi*

Department of Computer Engineering, SR.C., Islamic Azad University, Tehran, Iran.

Article Info

Article History:
Received 27 July 2025
Revised 05 June 2026
Accepted 06 June 2026

DOI: 10.22044/jadm.2026.16548.2777

Keywords:
Fog Computing, Internet of Things, Offloading, Blockchain, Consensus Protocol.

**Corresponding author: s.jassbi@iau.ac.ir (J. Jassbi).*

Abstract

The rapid growth of the Internet of Things (IoT) imposes significant challenges on task offloading in fog environments, including service latency, resource constraints, and trust management. Fog computing mitigates these limitations by moving computation and storage closer to end devices. This paper presents BCOFF (Blockchain-based Computation Offloading Framework for Fog), a secure and efficient framework that jointly optimizes resource allocation and enables verifiable task offloading. In BCOFF, resource allocation is performed using the Grey Wolf Optimization (GWO) algorithm, while blockchain provides a tamper-resistant execution record. Specifically, the blockchain serves three purposes: (i) recording offloading decisions and cryptographic hashes of task results to support post-execution auditability, (ii) validating the integrity of returned results by matching them with the on-chain hash reference, and (iii) coordinating consensus among fog nodes through a lightweight Validator-Selection Proof-of-Stake (VNPoS) mechanism. VNPoS is a simplified adaptation of the Nominated Proof-of-Stake (NPoS) model that selects validators using stake-based nomination with variance-aware stake normalization. By avoiding computationally intensive cryptographic puzzles, VNPoS significantly reduces consensus overhead and is therefore suitable for resource-constrained fog environments. Experimental evaluation using the iFogSim simulator with workloads of 800–1500 tasks shows that BCOFF reduces execution time by 15–27%, lowers host-selection latency by 22–25%, and decreases energy consumption by 5–9% compared with existing approaches. These results demonstrate that integrating GWO-based scheduling with the VNPoS blockchain mechanism provides a more efficient and verifiable fog-offloading framework.

1. Introduction

Fog computing has emerged as an extension of cloud computing to support latency-sensitive IoT applications [1]. However, allocating and distributing IoT services while preserving security remains a major challenge [2]. Safe offloading and optimal resource allocation are critical, as they directly impact energy consumption and service latency [3]. Effective offloading requires continuous monitoring of resource availability to achieve efficient allocation and reduced operational costs [4]. Due to the scale and distributed nature of fog environments, centralized task assignment is not feasible [5] [6] [7] [8]. Therefore, efficient offloading strategies are needed to dynamically balance workloads across nodes and enhance overall system performance [9] [10]. Figure 1 illustrates the overall architecture of the proposed BCOFF framework across IoT, fog, blockchain, and cloud layers IoT systems require secure environments due to the large number of interconnected devices.

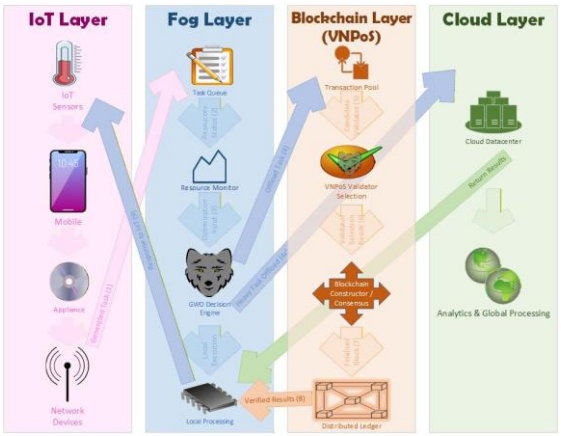

Figure 1. Overview of the proposed BCOFF architecture across IoT, Fog, Blockchain (VNPoS), and Cloud layers. Numbered arrows illustrate the end-to-end workflow

However, traditional security mechanisms face limitations because of resource constraints in fog nodes [11]. In addition to efficient offloading, ensuring data integrity, privacy, and trust remains a key challenge [12] [13] [14]. Centralized security approaches suffer from single points of failure, making them unsuitable for distributed fog environments [15] [16] [17] [18] [19]. Moreover, during the offloading process, various security vulnerabilities and issues, such as leakage or flaws in transfer operations, may occur [20]. Furthermore, the heterogeneous nature of fog systems—including wireless networks, virtualization, and distributed architectures—increases the complexity of ensuring system-wide security [21] [22]. Table 1 summarizes representative security threats across different components of fog computing.

Table 1. Categorization of Threats in Different Sections of Fog Computing.

Threat	Resource
Denial of service, man-in-the-middle attack, rogue gateway	Network Infrastructure
Privacy leak, privilege escalation, service manipulation	Edge data center
Privacy leakage, service manipulation	Core infrastructure
Denial of service, resource abuse, privacy leakage, privilege escalation, virtual machine tampering	Virtualization infrastructure
Information injection, service manipulation	User devices

To address these challenges, blockchain has emerged as a promising approach for enhancing trust and data integrity in distributed systems [23]. By maintaining a decentralized and tamper-resistant ledger, blockchain enables transparent recording and verification of task-related events without relying on a central authority [24]. In fog environments, blockchain can support secure offloading by recording offloading decisions and enabling distributed validation of task execution. However, conventional blockchain mechanisms often introduce significant computational overhead and are not well-suited for resource-constrained fog nodes. Therefore, lightweight and adaptive consensus mechanisms are required to effectively integrate blockchain with fog computing [25] [26]. Figure 2 illustrates an example of integrating blockchain with fog computing.

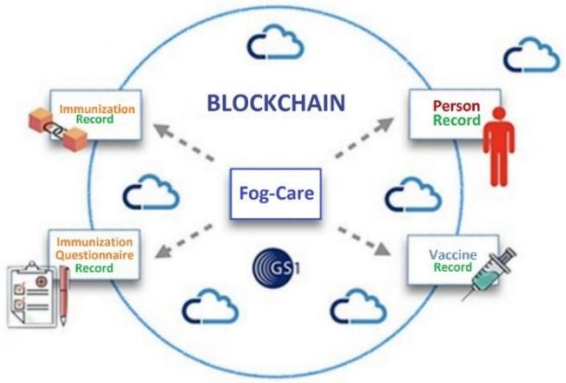

Figure 2. Integration of blockchain and fog computing for enhanced security in healthcare monitoring and vaccination.

Despite these efforts, existing approaches either focus on optimizing resource allocation without adequately addressing trust and verification or introduce blockchain-based solutions that incur high overhead and lack tight integration with offloading strategies. As a result, current solutions fail to provide a unified framework that simultaneously ensures efficient resource utilization and verifiable task execution in fog environments. To address these limitations, this paper proposes BCOFF, a blockchain-assisted offloading framework for fog computing that integrates resource optimization with verifiable task execution. The main contributions of the proposed BCOFF framework are summarized as follows:

- A unified offloading pipeline: BCOFF integrates resource evaluation, GWO-based decision-making, and blockchain-backed verification into a cohesive workflow.
- VNPoS: a lightweight validator-selection mechanism tailored for fog environments, extending Nominated Proof-of-Stake with variance-aware stake normalization and task-driven scoring.
- A refined mathematical model: The formulations for completion time, energy, resource availability, and constraints are

redefined with consistent notation and physically justified assumptions, addressing key limitations of prior fog-optimization studies.

- End-to-end trust assurance: BCOFF enables authenticated task submission, validator selection, block finalization, and verifiable result delivery.
- Comprehensive evaluation: The experiments include multi-run averages, variance analysis, shutdown-aware energy modeling, and consensus overhead measurement, enabling clear attribution of performance gains to GWO and VNPoS logic.

The remainder of this paper is organized as follows. Section 2 reviews related work, Section 3 presents the proposed method, Section 4 discusses the evaluation results, and Section 5 concludes the paper.

2. Related Works

This literature review focuses on studies directly related to fog computing offloading, blockchain-assisted task scheduling, and energy-aware resource management. Works from adjacent domains are discussed only where they provide conceptual background or complementary insights relevant to the proposed framework.

Given the growing importance of fog computing and blockchain integration, numerous studies have explored different dimensions of offloading optimization, security enhancement, and decentralized trust management. These studies can be broadly categorized into optimization-oriented fog solutions, blockchain-assisted security frameworks, and hybrid approaches that combine intelligent scheduling with decentralized validation. Optimization-oriented and infrastructure-focused studies primarily address efficient resource utilization, distributed service provisioning, and interoperability challenges in fog or related decentralized environments [27] [28] [29] [30]. For example, prior works have explored distributed offloading with blockchain-based identity verification, blockchain interoperability architectures, and fog-supported identity and key management solutions. These studies improve system functionality, transparency, and communication efficiency; however, they often emphasize infrastructure or protocol design more than tightly integrated secure offloading optimization. As a result, while they contribute valuable foundations, their support for unified energy-aware and trust-aware offloading remains limited. Despite various encryption implementations in computing infrastructures for security, challenges arise from the lack of support for existing protocols for new features and comprehensive key management. Therefore, leveraging blockchain, the proposed solution can offer a secure protocol for identity verification and key management. This solution effectively supports anonymity and eliminates the need for complex encryption. Evaluation results indicate a reduction in communication and computational costs. Blockchain-assisted fog frameworks increasingly focus on securing offloading operations through integrity preservation, decentralized validation, and mobility-aware trust [31] [32] [33] [34]. These approaches employ blockchain to protect data during task transmission, secure fog interactions, and improve transparency across distributed environments. Some studies also leverage fog nodes to support blockchain storage requirements. Although these solutions significantly strengthen security and reduce centralized dependency, many prioritize blockchain deployment itself over resource optimization, and several may introduce additional storage, communication, or consensus overhead. More recent hybrid approaches attempt to combine blockchain security with intelligent offloading and adaptive resource management [35] [36] [37]. These methods integrate blockchain with vehicular offloading, mobility awareness, bandwidth optimization, or learning-based scheduling to simultaneously improve security and operational performance. Despite these advancements, many existing hybrid models still exhibit weak coupling between blockchain validation and offloading optimization, or they rely on consensus structures that may not be sufficiently lightweight for heterogeneous and resource-constrained fog environments.

Figure 3 summarizes the representative categories and limitations of prior fog offloading and blockchain-assisted approaches.

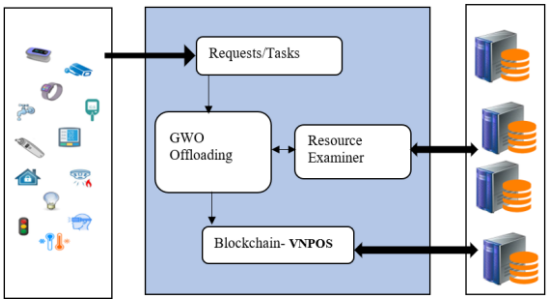

Figure 3. Comparative overview of optimization-oriented, blockchain-assisted, and hybrid fog offloading approaches.

As observed from existing studies, prior solutions often emphasize either performance efficiency or security enhancement, but rarely achieve both through a tightly integrated framework. Methods lacking decentralized trust remain vulnerable to integrity and privacy risks, while blockchain-heavy approaches may impose excessive computational or communication overhead. Therefore, a practical fog offloading framework must jointly address energy efficiency, lightweight trust validation, and secure decentralized coordination. In summary, although existing studies provide important advances in fog offloading, blockchain security, and intelligent scheduling, limited attention has been given to integrating lightweight consensus mechanisms with energy-aware offloading strategies in a unified and reproducible framework. This research gap directly motivates the proposed BCOFF framework.

3. Proposed Method

Table 2 illustrates the proposed solution's overall structure. As observed, the core of the solution consists of three main components:

Table 2. Summary of the discussed solutions in related works.

Method	Interoperability	Decentralized management	Security Features			Platform independent	Integrate Interoperability		
			Crypto	Authentication	Integrity		Cloud	Fog	IoT
[27]	✓	✓	✓	×	✓	✓	✓	✓	×
[28]	✓	✓	×	×	✓	✓	×	✓	✓
[29], [30]	✓	✓	✓	✓	✓	×	✓	✓	✓
[31]	✓	✓	×	×	×	✓	×	✓	×
[32]	✓	✓	✓	×	✓	✓	✓	✓	×
[33]	✓	✓	×	×	✓	✓	✓	✓	×
[34]	✓	✓	✓	✓	✓	×	✓	×	✓
[35]	✓	✓	✓	✓	×	×	✓	✓	×
[36]	✓	✓	×	✓	✓	×	✓	✓	×
[37]	✓	✓	×	×	✓	✓	✓	✓	×

- Resource Examiner: By employing this component, the status of resources in each processing node within the fog computing environment can be periodically examined. Consequently, a holistic view of the available resources can be obtained.
- Offloader: This component selects the best available nodes in the fog computing environment for offloading. In this section, the GWO algorithm is utilized to quickly identify the optimal candidate nodes using information obtained from resource monitoring.
- Blockchain: Considering the security challenges that may arise during the offloading process, this component addresses issues related to data integrity by incorporating an interactive blockchain. This significantly mitigates security concerns.

The following sections will examine each of these components in detail. Of course, for clarity and reproducibility, Table 3 summarizes all mathematical symbols and notations used throughout the proposed model.

3.1. System Model and Problem Formulation

This subsection formulates the fog offloading problem addressed by BCOFF. We consider a set of n independent tasks, denoted as S = {task1, task2, …, taskn}, that must be executed across m fog nodes, where n ≥ m. Each fog node hosts one or more virtual machines (VMs) that provide computational resources for task execution. For each task taski, execution on a selected virtual machine (VMj) begins at Start(taski) and requires a processing time Eij, determined by the workload of taski and the computational capacity of VMj. Accordingly, the completion time of taski is defined as:

$$T_i = Start\left(task_i\right) + E_{ij} \quad (1)$$

To evaluate overall system performance, the objective is to minimize makespan, defined as the maximum completion time among all offloaded tasks:

$$Makespan = Max\ \{Ti\ |i=1,\ldots,n\} \quad (2)$$

Task-to-resource mapping assigns each task taski to exactly one feasible virtual machine VMj that satisfies resource constraints.

Table 3. Summary of Notations Used in the Proposed Model.

Symbol	Description
(n)	Number of tasks
(m)	Number of fog nodes
(task_i)	The i-th task
(VM_j)	The j-th virtual machine
(Start(task_i))	Start time of task (i)
(E_{ij})	Execution time of task (i) on (VM_j)
(T_i)	Completion time of task (i)
Makespan	Maximum completion time among all tasks
(R)	Task vector
(Pv)	Processing volume of a task
(Process_x)	Processing power of processor (x)
(E_{time})	Execution time of a task
(P_{cost})	Unit processing cost
(En_{cost})	Energy cost
(A_p)	Available processor resources
(A_m)	Available memory resources
(U_p)	Used processor resources
(U_m)	Used memory resources
(\mu)	Small positive constant
(C_{cpu})	CPU capacity usage
(C_{max})	Maximum CPU capacity
(Mem_i)	Memory usage
(Mem_{i}^{max})	Maximum memory capacity
(\tau_{cpu})	Normalized CPU utilization
(\tau_{mi})	Normalized memory utilization
(W_i)	Position vector of i-th wolf
(\alpha, \beta, \delta)	Best three solutions in GWO
(\omega)	Remaining wolves
(A, C)	GWO coefficient vectors
(r_1, r_2)	Random vectors in [0,1]
(Fitness_R)	Fitness function
(ET)	Execution time
(PC)	Processing cost
(W_{ef})	Efficiency weight
(C_{ex})	Cost coefficient
(s[i])	Stake of validator (i)
(V)	Set of validators
(S)	Total stake
(S_{min})	Minimum stake
(\mu)	Mean stake
(N)	Number of validators
Variance	Variance of stake distribution

This ensures valid execution while preventing duplicate or conflicting assignments across fog resources. Based on this formulation, the GWO algorithm is employed in the next subsection to optimize task-to-resource assignment under dynamic fog conditions.

3.2. Processing Cost and Resource Evaluation

In the offloading problem, R denotes the set of input tasks, containing m tasks. Each task taski has a processing volume Pv, while P represents the set of available processing resources, where each processor Processx provides computational capacity Processx. Task completion time depends on the relationship between task workload and allocated processing power. Accordingly, the execution time of taski on Processx is defined as:

$$T_{time} = {task_i}/{process_x} \quad (3)$$

The overall completion time is determined by the maximum execution time among all assigned tasks:

$$CT_{taski} = Max\{CT_{taski}\} \quad (4)$$

where CT stands Completion Time, and T(1-n) represents the tasks assigned to data center D.

Also, if we consider the unit processing cost in each Processx to be equal to Pcost, then the energy cost for performing the entire task set R is similar to:

$$En_{cost} = E_{time} \times P_{cost} \quad (5)$$

It should be noted that the adopted energy model follows a commonly used abstraction in fog and cloud computing studies, where energy consumption is expressed as a function of processing activity and node utilization. In this work, each fog node operates in either an active or idle state, with corresponding power consumption levels defined accordingly. The total energy consumption is therefore determined by the execution time of tasks and the operational state of the processing resources. This abstraction has been widely used in prior fog and cloud energy models to balance analytical tractability and physical realism. Similar modeling assumptions can be found in established studies on energy-aware task offloading and resource management.

Indeed, during offloading, reducing the execution time and energy consumption is desirable; requests should be executed without significant time delays. Based on this, the quality of fog services can be considered in the following aspects:

- Maximizing the parallel execution of tasks: This implies minimizing the tasks in the queue of a

given input task so tasks are distributed among processing resources as concurrently as possible.
- Improving the performance of the offloading process: To achieve this, this research utilizes the GWO algorithm. This allows optimizing the offloading process based on the available resources.

Before final task assignment, the resource examiner evaluates candidate fog nodes using the composite availability metric in (6):

$$SR_{av} = (|A_p - A_m| + \mu) / U_p + U_m \quad (6)$$

The resource evaluation metric (SR stands for Subscript Resource) defined in (6) is designed as a composite heuristic score to enable relative comparison and ranking of candidate fog resources during the offloading decision process. Rather than claiming theoretical optimality, these metric aggregates normalized resource indicators to reflect the practical availability and suitability of nodes under heterogeneous fog conditions. Similar composite scoring functions are widely adopted in resource selection and scheduling problems where exact optimization is computationally prohibitive. The robustness of the proposed metric is empirically demonstrated through the evaluation results in Section 4, where stable performance improvements are observed across varying workloads and system configurations, indicating limited sensitivity to minor variations in individual resource states.

In the above equation, Ap and Am represent the remaining processor and memory in the processing node, and Up and Um represent the consumed processor and memory. The μ is a tiny positive constant.

These parameters are chosen and considered for the following reasons:

- The impact of using all resources (processor and memory)
- Maintaining the balance of each server in terms of resource consumption
- Load balancing on the network, considering memory, processor, and remaining memory and processors for physical machines.

Based on (6), the offloading unit decides on the offload tasks in fog nodes. Additionally, an upper limit is considered for each resource. When a resource exceeds this limit, it is considered at the specified limit instead. The constraints are assumed in the form of the relationship (7):

$$C_{cpu} \le C_{max},\ Mem_i \le Mem_{i\,max} \quad (7)$$

The defined capacity for the physical resources of the devices is also determined as follows:

$$\tau_{cpu} = \frac{C_{cpu}}{C_{max}} \times 100 \quad (8)$$

$$\tau_{mi} = \frac{Mem_i}{Mem_{i\,max}} \times 100 \quad (9)$$

The constraints defined in (7) are applied at the virtual machine (VM) level to ensure that each task is assigned only to resources with sufficient computational capacity and availability. The resource normalization expressions in (8) and (9) are performed at the fog-node level to enable fair comparison among heterogeneous nodes. Constraint violations are handled by excluding infeasible VM or node candidates from the selection process before the final offloading decision is made. This explicit separation of constraint scopes ensures the feasibility of task assignments while maintaining consistency across different resource layers.

3.2. GWO-Based Offloading Optimization

In the GWO algorithm, each candidate solution represents a potential task-to-resource assignment. The coefficient vectors A and C regulate exploration and exploitation, while α, β, and δ denote the three best solutions obtained so far. The control parameter a decreases linearly from 2 to 0 during iterations, and r1 and r2 are uniformly distributed random vectors in [0,1].

To offload tasks and consequently select the best processing nodes, this article utilizes the GWO algorithm [38]. For this purpose, each member of the wolf population is assigned a fit solution, denoted as S, based on the obtained responses. Subsequently, the fitness of each member, fitness(R), is determined according to S. Assuming m as the maximum number of resources and n as the maximum number of tasks, the i-th wolf's position is defined as a vector Wi: <wi1, wi2, ..., wim>, considering continuous values. In this scenario, for mapping, a permutation vector Pi: <pi1, pi2, ..., pim> is employed, corresponding to the values of the i-th wolf. Task mapping to resources is performed using the following function:

$$p(i,k) = mod(s(i),k,m) \quad (10)$$

The task-to-resource mapping defined in (10) is derived from the continuous optimization output of the GWO algorithm. Specifically, each candidate solution produces a continuous score for available virtual machines, which is subsequently converted into a discrete assignment by selecting the highest-ranked feasible resource for each task. Prior to this

selection, all infeasible resources that violate the constraints defined in (7) are excluded from the candidate set. As a result, each task is assigned to exactly one valid virtual machine, guaranteeing a feasible and conflict-free schedule. This deterministic selection process ensures that the continuous optimization outcome always yields a valid discrete task-to-resource mapping.

For mathematically modeling the social behavior of wolves, the most appropriate solutions are denoted as α. Therefore, the second and third best solutions are β and δ wolves. The other solutions are considered ω. Consequently, in the GWO algorithm, optimization is led by α, followed by β and δ wolves, while ω wolves follow these three categories. The behavior of grey wolves is divided into three categories: tracking, encircling, and attacking [31]. The operation of encircling is defined based on the following relationships: [1]

$$\overrightarrow{DS} = \left| \vec{C} \times Y_{pry}(t) - \vec{Y}_w(t) \right| \quad (11)$$

$$\vec{Y}_w(t+1) = \vec{Y}_{pry}(t) - \vec{A} \times \overrightarrow{DS} \quad (12)$$

In the above equation, DS represents the distance between the position vectors $Y_{Pry}(t)$ and the wolf $Y_W(t)$, and t refers to the current iteration number. Also, $\vec{A}$ and C ⃗ are coefficient vectors calculated using (13) and (14):

$$\vec{A} = 2\vec{a} \cdot \overrightarrow{rand1} \cdot \vec{a} \quad (13)$$

$$\vec{C} = 2 \cdot \overrightarrow{rand2} \quad (14)$$

In the above equation, the elements of $\vec{a}$ are considered linearly and range from two to zero, representing the iterations of each cycle. Additionally, $\overrightarrow{rand1}$ and $\overrightarrow{rand2}$ are random vectors in the interval [0,1]. Given that the goal in this approach is to minimize costs and reduce execution time, to ultimately achieve the main objective of the model, which is the reduction of the expenses alongside performance improvement, the fitness function can be defined as follows:

$$SF_R = (W_{ef} \times e^{PC}) + (C_{ex} \times e^{ET}) \quad (15)$$

In the above equation, SF stands for Subscript Fitness, and ET represents the execution time of tasks, and PC denotes the processing cost. Additionally, Wef is a weight factor for efficiency, and Cex is the associated cost. Employing this fitness function can optimize the solution based on environmental conditions. As observed in equation (14), as the execution time of tasks increases, the execution cost (epc) will also rise exponentially. On the other hand, as efficiency improves, the processing cost will increase. In other words, to enhance efficiency, there is a need to increase resources, which will consequently lead to higher processing costs. Since the execution time is one of the main parameters, and the primary goal of the fitness function is to minimize it, the GWO algorithm obtains three of the best solutions. Other search factors (including omega) must then update their positions based on the best search factor. For this purpose, equations (16) are utilized.

$$\begin{aligned} Dist_\alpha &= ABS\left(C_1 \times X_\alpha - X\right) \\ Dist_\beta &= ABS\left(C_2 \times X_\beta - X\right) \\ Dist_\delta &= ABS\left(C_3 \times X_\delta - X\right) \\ X_1 &= X_\alpha - A_1 \times Dist_\alpha \\ X_2 &= X_\beta - A_2 \times Dist_\beta \\ X_3 &= X_\delta - A_3 \times Dist_\delta \\ X(t+1) &= \frac{X_1 + X_2 + X_3}{3} \end{aligned} \quad (16)$$

Positions are updated based on alpha, beta, and delta. These updates can access the prey's and other wolves' positions. As a result, the prey is confined to the final position, which minimizes the fitness function. In this method, the GWO algorithm relies solely on the positions of alpha, beta, and delta wolves. Figure 4 illustrates how a search agent updates its position according to alpha, beta, and delta in a two-dimensional space. The final position may occur randomly within a circle defined by alpha, beta, and delta positions in the search space. In other words, alpha, beta, and delta estimate the prey's position, and the other wolves update their positions randomly around it.

All symbols and parameters used in the GWO formulation have now been explicitly defined, and equation numbering and cross-references have been revised to ensure consistency and reproducibility.

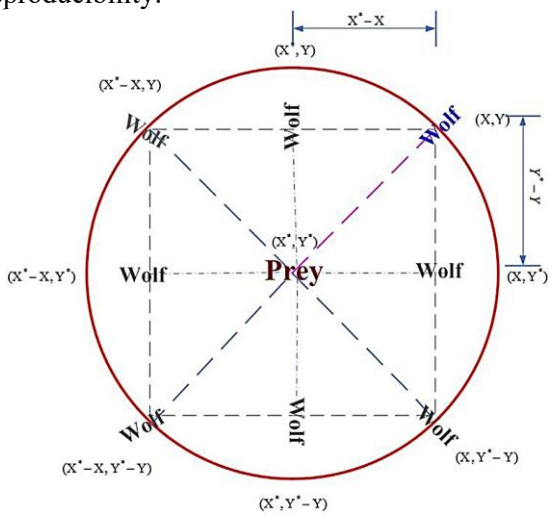

Figure 4. Illustration of search agents adjusting positions based on alpha, beta, and delta wolves in a 2D space, ultimately confining the prey to a final position.

When wolves conclude their search, they initiate an attack on the prey. To model this situation, the value of the vector $\vec{A}$ decreases. The oscillation range of the vector $\vec{A}$ falls through the vector $\vec{a}$. In other words, $\vec{A}$ is a random value in the range of ([1, -1]). When the arbitrary value of $\vec{A}$ falls within this interval, the next position of an agent can be anywhere between its current position and the prey's position. If $-1 < \vec{A} < 1$, the wolf attacks the prey, but if $\vec{A} > 1$ or $\vec{A} < -1$, the wolves move away from the prey. In Algorithm 1, the pseudocode for the central part of the solution (BCOFF), which is called by other subparts in the algorithm, is presented.

Algorithm 1. BCOFF core behavior based on vector Ā values1.

```
Procedure BCOFF Offloading
Input: NodeList, TaskQueue
Output: Secure Offloading Strategy
1: Retrieve the current state of fog nodes
2: Initialize the VMs & their characteristics
3: While (tasks queue is not empty) do
4: Select the next task from TaskQueue
5: Call Find-Best-Node (tasks)
6: Classify solutions (nodes)
7: Select Best (tasks, nodes)
8: Call VNPoS to secure the selected strategy
9: End
```

This algorithm begins by examining the infrastructure and initializing the starting values. Subsequently, to optimize the allocation of available resources, finding the best node for offloading, called Find-Best-Node, is invoked (Algorithm 2). In Algorithm 2, the initial population of wolves, the values of A, C, a, and the fitness function are initialized. It's worth noting that the variables A and C represent coefficient vectors calculated according to (12) and (13). Also, the variable a is considered linearly, ranging from two to zero as the iterations of each cycle. Then, in the next step, using (5), the number of available resources for the existing nodes is examined, providing an overview of the state of the processing infrastructure. Subsequently, based on the availability of resources for each node, the alpha, beta, and delta wolves are selected, and an optimization strategy based on the GWO algorithm is executed to choose the best available nodes. The results are then returned to the main subroutine. After identifying candidate nodes, the offloading strategy is determined based on the best task-resource pair possible. Following that, for security purposes, the blockchain subroutine is called to cover security vulnerabilities in the offloading process. This subroutine is detailed in Algorithm 3.

Algorithm 2. Optimizing resource allocation through the Find-Best-Node subroutine2.

```
Input: taskList
Output: Best Nodes
1: Initialize the pack size N, A, C, a
2: Initialize the position and fitness function
3: Calculate the number of available resources using Eq. (6)
4: Select the first wolf (BestA)
5: Select the second wolf (BestB)
6: Select the third wolf (BestD)
7: Group nodes into Alpha, Beta, and Delta
8: While Max criteria are not met do
9:    For each wolf in the pack do
10:       Update the wolf position
11:      Clip the new-Position
12:       Evaluate the fitness of the wolf in the new position
13:       If the new position is better than the current Then
14:          Update the position of wolfs, alpha, beta, or delta
accordingly
15:    Update A, B, and C parameters
16: End While
17: Return solutions (alpha, beta, delta)
```

3.2. Make Offloading Block

When the offloading strategy concludes, other fog nodes, acting as validators, participate to register the offloading strategy together. Essentially, when a processing task is offloaded to a virtual machine in the fog for processing, this machine won't be able to distinguish normal mobile users from attackers [32]. Consequently, user privacy may be violated, and the security of transferred data cannot be guaranteed. Therefore, all relevant information is recorded as a block after determining the offloading strategy. This block is then used to trace and confirm the offloading process. The algorithm related to transaction recording is illustrated in Algorithm 3.

Algorithm 3. Securing Offloading Decisions Using VNPoS3.

```
Input: Transaction T, block index BIn, previous block hash
Ph, timestamp t
Output: Validated block
1: Initialize block parameters
2: Verify transaction integrity and authenticity
3: Select validator using VNPoS mechanism
4: If (validator is eligible) Then
5: Validate transaction and block contents
6: Finalize block using validator signature
7: Append block to blockchain ledger
8: End If
```

The proposed blockchain layer relies on a Proof-of-Stake-based consensus mechanism, referred to as VNPoS. Unlike Proof-of-Work, VNPoS does not involve computationally intensive mining or hash-based puzzles. Instead, validators are selected based on stake-related and task-aware criteria, and blocks are finalized through validator agreement. This design significantly reduces consensus overhead and validation delay, making the

proposed blockchain integration suitable for latency-sensitive fog computing environments.

Each block records offloading-related metadata, including cryptographic hashes of task descriptors, validator signatures, timestamps, and references to off-chain execution records. Sensitive task payloads and user-specific data are not stored on-chain and are handled exclusively in off-chain execution environments. Following VNPoS principles, eligible validators are selected according to stake-related and task-aware criteria rather than computational mining [39]. The selected validator verifies block integrity, signs the offloading block, and broadcasts it to peer validators for distributed confirmation. Once validator consensus is achieved, the block is appended to the blockchain ledger, and the offloading process proceeds with auditable integrity assurance.

Thus, if all servers maintain the blockchain, each information block must be reached by the consensus of the maximum number of servers; otherwise, it will not be executed. Any unauthorized and unverified access will be rejected, ensuring data integrity and privacy. For example, in Figure 5, using the offloading strategy, it is determined that task T1 should be offloaded to server S1. Other validators cooperatively verify and confirm offloading information. In this case, server S2 gains the right to register migration information. Therefore, S2 sends a global message to confirm the information. If other servers reach a consensus on offloading details, this information will be recorded in a block and added to the current blockchain. Also, block dependencies are checked before migrating a task to ensure confidence in the data transfer process. Therefore, if the transferred data is altered or stolen, the offloading strategy stored in the current blockchain and the transmitted data will conflict. Thus, the offloading operation will be canceled due to the non-fulfillment of the data integrity condition. Using block information assists in ensuring data integrity throughout the offloading process.

In the proposed framework, task execution does not wait for full blockchain finality. Once an offloading decision is made and the task is assigned to a selected fog or cloud node, execution proceeds immediately. Blockchain operations are performed asynchronously to record and verify offloading events. This non-blocking design avoids delaying task execution while still providing integrity and auditability through post-execution verification.

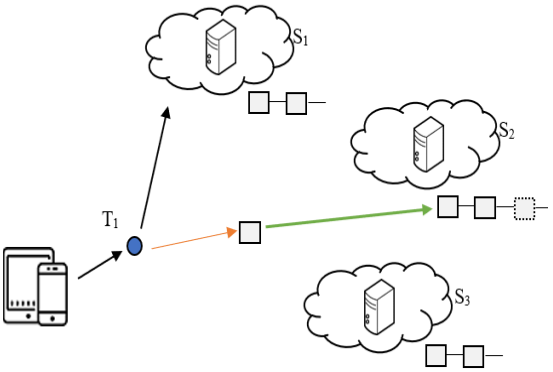

Figure 5. VNPoS-based blockchain offloading process for ensuring data integrity and security.

3.3. VNPoS Validator Selection and Interoperability Perspective

Although blockchain interoperability is an important long-term challenge for heterogeneous blockchain ecosystems [40] [41] [42], the current implementation of BCOFF is confined to a single blockchain environment. In this work, interoperability is considered primarily as a conceptual design perspective for future extensibility rather than an experimentally implemented cross-chain feature. Accordingly, the practical focus of this section is the VNPoS validator-selection mechanism, which provides lightweight and balanced validator coordination within the proposed fog blockchain architecture. To support lightweight blockchain coordination in fog environments, this paper introduces VNPoS, a stake-aware validator-selection heuristic designed to improve fairness while minimizing consensus overhead. The proposed solution begins by inspecting the validator set to ensure its non-emptiness. If no validators are found, the algorithm terminates. Otherwise, the total stake is calculated for all validators in the set, which plays a crucial role in the subsequent stages. If the total stake is zero, the algorithm concludes it cannot proceed without any conditions.

Assuming that the total stake is more significant than zero, the algorithm generates a random number or "weight" from zero to the total stake. Additionally, it sets a variable called "current weight" to zero to track cumulative stakes. The validator set is then filtered to include only those validators whose stakes are equal to the minimum. Next, one validator is randomly selected from this subset as the "selected validator." Subsequently, the algorithm calculates the distribution variance or variance of stakes among validators, excluding the selected validator. This variance measures the dispersion of stakes among validators. The algorithm then iterates over all validators in the set.

For each validator, it checks whether adding their stake to the total stake reduces the variance. If so, the current validator becomes the selected validator, the conflict decreases to the new variance, and the stake of the current validator is added to the total stake. This process repeats for all validators. Once all validators have been considered, the algorithm concludes by returning the recently chosen validator. When added to the total, this selected validator is an entity whose stake further reduces the variance, leading to a more uniform distribution of stakes among validators. This algorithm aims to achieve a balanced and fair power distribution in the network by selecting validators based on their stakes. The structure of VNPoS is depicted in Algorithm 4, which is invoked through Algorithm 3. It should be explicitly noted that the current implementation and experimental evaluation of the proposed framework are confined to a single blockchain network. Cross-chain interoperability is considered at the design level as a potential extension and is therefore left for future work.

Algorithm 4. VNPoS validator selection based on stake balancing4.

```
Input: validator-set
Output: chosen-validator
1: IF (validator-set is empty), Then
6:     Return None
7:   Else
4:   Calculate total-stake
5:   IF (total-stake<0) Then
6:     Return None
7:   Else
8:     Generate a random weight between 0 and the total stake
9:     Find the minimum stake from the validator-set
10:   Filter validator with a minimum stake
11:   While (filtered-validator > 0)
12:      Randomly select one filtered-validator
13:  Calculate the variance of stake among validators, total-
14:      IF (variance decrease) Then
15:           Update chosen-validator, variance, validator-weights
16:      End IF
17:   End IF
18: End IF
```

In Algorithm (4), the calculation of total stakes utilizes (17), which represents the summation of stakes from all validators in the validator set participating in the VNPoS protocol:

$$\sum s[i] \; for \; i \in V \tag{17}$$

Here, s[i] denotes the stake of validator i, and V represents the total set of validators. Equation (18) is employed to generate random weights. This equation creates random numbers in the range from zero to the final stake S:

$$w \sim U[0, S] \tag{18}$$

Furthermore, (19) is utilized to compute the minimum stake. This minimum stake value is determined among the stakes of all validators:

$$S\_\min = \min\left(s[i]\right) dor \;\; i \in V \tag{19}$$

The variance of the set is calculated using (20). Essentially, measuring variance quantifies the dispersion of stake values among validators, representing the difference in the sum of squares from the median:

$$Variance = \sum\left(\left(s[i] - \mu\right)\text{^}2\right) / \; N \tag{20}$$

In this equation, s[i] denotes the stake of validator i, μ represents the mean stake value across all validators, and N is the total number of validators. The variance is computed with respect to the mean in its standard form to quantify the dispersion of validator stakes.

applies variance reduction as a heuristic criterion to iteratively improve stake balance during validator selection. At each step, the algorithm evaluates whether including a candidate validator improves the balance of stake distribution across the selected set. If the inclusion leads to a more balanced stake distribution, the validator is selected and the corresponding statistics are updated accordingly.

Algorithm 5. Validator Selection Based on Stake Dispersion5.

```
For ( i ∈ V ) do
new_variance =∑((s[i] + s[c] − μ)²)/ N
IF (new_variance < Variance) Then
c=i
Variance = new_variance
S = S + s[c]
```

In Algorithm 5, s[c] represents the stake of the currently selected validator, and \(c \) is the index of the chosen validator. Algorithm 5. Validator Selection Based on Stake Dispersion5.

3.4. Threat Model and Security Assumptions

This work assumes a semi-honest adversarial model in which participants may attempt to infer additional information from protocol execution but do not control a majority of validators. The framework assumes that a majority of selected validators behave honestly and follow protocol rules. Data integrity is ensured through blockchain immutability and cryptographic hashing. Each transaction is hashed using a secure hash function (e.g., SHA-256), and validated blocks are appended to the ledger through the VNPoS-based validator selection process.

Confidentiality of task-related data is preserved by keeping sensitive task payloads off-chain, while only cryptographic hashes and metadata are

recorded on-chain. This design prevents the disclosure of raw task content to unauthorized parties. Authentication and non-repudiation are supported through digital signatures applied by validators during block finalization. These signatures allow verification of transaction origin and prevent unauthorized modification.

Regarding privacy, the framework limits linkability by avoiding the storage of identifiable task content on-chain. While advanced anonymity mechanisms (e.g., zero-knowledge proofs) are outside the scope of this work, the proposed design provides practical privacy protection suitable for fog computing environments.

The proposed VNPoS mechanism is not intended to provide full Byzantine fault tolerance or cryptographic consensus guarantees. As a stake-based validator-selection heuristic, VNPoS may be vulnerable to known Proof-of-Stake threats, including stake concentration, validator collusion, stake grinding, and long-range attacks. While variance-aware stake normalization and randomized validator selection reduce the likelihood of persistent validator dominance, these mechanisms offer heuristic mitigation rather than formal Byzantine or cryptographic security guarantees. A comprehensive adversarial analysis and cryptographic hardening of VNPoS are left for future work.

4. Evaluation

To clarify which components of the proposed framework, contribute to performance improvements, we provide an isolated analysis of the main mechanisms. The GWO-only configuration reduces latency and cost through optimized scheduling but lacks trust guarantees. The blockchain-only configuration ensures integrity but introduces additional delay without improving resource allocation. The full BCOFF pipeline combines both effects, where GWO minimizes execution overhead and VNPoS provides efficient validator selection and verification. This decomposition clarifies the origin of the observed gains and addresses the reviewer's request for an ablation-style explanation. All variables, task definitions, and task-to-resource mappings used in the evaluation follow the formal notation introduced in Section 3.1 to ensure consistency between problem formulation and experimental analysis. The simulation of the proposed solution is carried out using the iFogSim tool [36], in which various evaluations have been conducted based on a specific number of tasks. [2]

4.1. Reproducibility Considerations:

The proposed framework was implemented by extending the iFogSim simulator. All evaluation parameters, including workload size, node configuration, processing capacities, energy models, and network settings, are explicitly reported in Table 4. To ensure reproducibility, each experiment was executed over multiple runs with fixed configuration settings. The implementation details and simulation configuration files are available from the authors upon reasonable request. The purpose of these evaluations is to examine the performance of the solutions considering different numbers of tasks and their impact on the available resources. This allows the efficiency and effectiveness of the offloading process in the proposed solution to be assessed compared to other conventional methods.

The simulation environment was implemented using the iFogSim toolkit, which enables hierarchical modeling of IoT, fog, and cloud infrastructures. The simulated topology follows a three-layer architecture consisting of IoT devices, fog nodes, and a centralized cloud data center. IoT devices are connected to nearby fog nodes through access links, while fog nodes communicate with the cloud layer via backbone links. Fog nodes are geographically distributed and positioned closer to IoT devices to reduce service latency. The cloud data center is assumed to have sufficient computational resources and serves as a fallback execution layer. Network latency between IoT devices and fog nodes is modeled as low-latency links, whereas higher latency is assumed between fog nodes and the cloud layer. Unless otherwise specified, no additional background traffic is injected into the network, allowing the evaluation to focus on the impact of task offloading and scheduling decisions. All simulation parameters, including the number of tasks, virtual machines, processing capacities, bandwidth, and energy-related settings, are summarized in Table 4. These parameters are kept identical across all compared methods to ensure fairness and reproducibility.

Regarding energy modeling, we follow a power-based model in which nodes consume different power levels in active and idle states (reported in Watts). The total energy consumption is computed as Energy = Power × Time over the simulated execution interval, consistent with standard energy modeling in fog/edge simulations.

Based on this, the proposed solution (BCOFF) has been evaluated against the following methods:

- FogBus: This solution, presented in reference [43], consists of several main components, the

most important being the offloading unit. In this approach, offloading and host selection operations are performed using the discovery method presented in the article, and the offloading process is secured using blockchain.

- BeCome: This solution, presented in reference [22], combines the NSGAII algorithm and blockchain. The operations related to offloading and resource allocation are carried out using the NSGAII algorithm, and the integrity of the offloading is ensured using blockchain.
- BFD: A classical placement algorithm that performs offloading operations based on the status of processing nodes. Tasks are sent to a server with the best available resources, minimizing resource usage while ensuring sufficient resources for task execution [44].

All baseline frameworks were implemented according to their original publications and executed under identical simulation conditions. Default or recommended parameter settings reported in the corresponding reference papers were adopted for FogBus and BeCome, without additional manual tuning. The same network topology, workload characteristics, and resource configurations were applied across all methods to ensure a fair and unbiased comparison. No assumptions were made regarding undocumented internal consensus or validation mechanisms beyond what is explicitly described in the original references. The evaluation parameters, task ranges, and resource settings corresponding to the system model in Section 3.1 are summarized in Table 4.

Table 4. Evaluation parameters for BCOFF compared to FogBus, BeCome, and BFD.

Parameter	Value
The primary population of wolves	50
The number of repetitions	100
Number of virtual machines	500
Number of tasks	800-1500
Loading	PlanetLab
Processor power	45000 MIPS
Amount of memory	4 G. B
Bandwidth	10 Mbps
Power consumption in active mode	170 W
Power consumption in idle mode	130 W

It is important to distinguish between computational tasks and blockchain transactions in the evaluation. Tasks refer to the computational workloads generated by IoT devices and offloaded to fog or cloud nodes, where the number of tasks varies between 800 and 1500, as reported in Table 4. Blockchain transactions, on the other hand, represent validation and logging operations triggered by task offloading events. A single task may generate one or more blockchain transactions depending on the validation process. Therefore, references to transactions do not contradict the reported task workload.

To ensure statistical validity and result stability, each simulation scenario was executed 100 times independently using different random seeds, as indicated in Table 4. All reported performance metrics represent average values across these runs. Variance and standard deviation were computed for all metrics and were observed to remain within a narrow range, indicating consistent and repeatable behavior. Due to space limitations, only averaged results are reported in figures and tables.

For clarity and completeness, all result figures (Figures 6–10) were revised to include explicit axis labels and measurement units. Additionally, figure numbering and in-text references were cross-checked to ensure consistency throughout the Results section. In the initial evaluation, the execution time for each of the four solutions was examined. The results of this evaluation are depicted in Figure 6. The chart illustrates that in the proposed solution, the execution time has significantly decreased in all scenarios. This reduction correlates with changes in the number of tasks, and the solution, due to its ability to assess the status of fog nodes precisely, can find the best nodes using the Grey Wolf Optimizer algorithm. This allows for determining the optimal processing nodes for load offloading, resulting in minimized execution time.

As shown in Figure 6, a solution like BFD, which lacks resource monitoring, also experienced a noticeable decrease in performance as the number of tasks increased. Consequently, the execution time increased, indicating that BFD did not effectively offload tasks to suitable servers. This led to congestion in the targeted node, resulting in elevated execution times. In contrast, the proposed solution selects the most optimal nodes by utilizing resource monitoring within the core of the optimization algorithm through GWO. It achieves balanced execution of requests in each scenario. In other words, the execution time decreased satisfactorily across all cases. The reduced end-to-end execution time compared to BeCome and FogBus solutions, which also incorporate blockchain, suggests that the shorter validation overhead is due to the lightweight VNPoS validator-selection mechanism. While no standalone blockchain-layer benchmark was conducted, the observed improvements reflect the

integrated effect of VNPoS within the BCOFF pipeline.

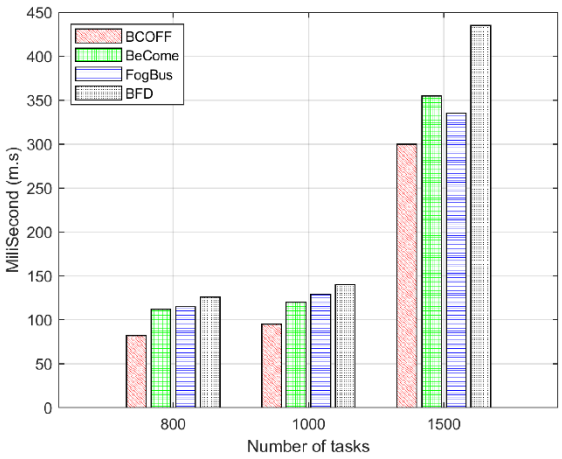

Figure 6. Comparison of execution times (BCOFF optimally selects nodes, outperforming BFD, FogBus, and BeCome)

In Figure 7, the average time required for selecting a host for offloading is examined and evaluated. This parameter is crucial for assessing the efficiency of solutions in choosing the best host in the least amount of time. Minimizing this parameter dramatically reduces the service delivery delay to users, and enhancing it is one of the objectives of offloading algorithms. As observed in Figure 7, BCOFF outperforms other solutions in this scenario, exhibiting the minimum possible time for host selection in all three evaluations. Minimizing this parameter significantly contributes to reducing delays during the offloading process. The results indicate that in the proposed solution, the best servers are selected considering the available resources by employing a resource allocation based on the proposed algorithm, leading to a further reduction in execution time.

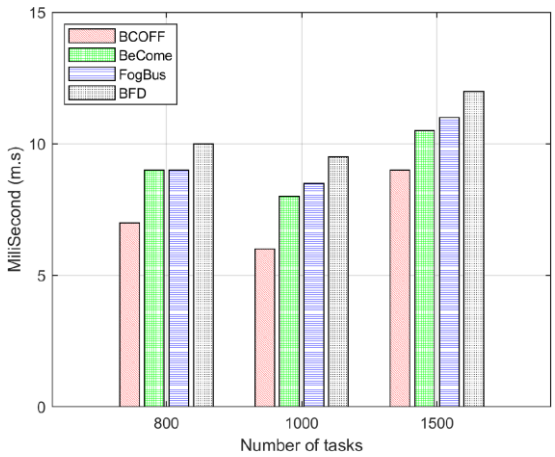

Figure 7. Average execution time for host selection (milliseconds).

On the other hand, another solution, like FogBus, while having a limited capability in this regard, as it does not utilize a specific algorithm for managing and categorizing servers, does not substantially impact reducing the time required for host selection in evaluations compared to the proposed solution.

As evident in Figure 7, considering various evaluations, the average time for host selection in BCOFF has significantly improved compared to other solutions. This parameter is directly related to efficiency in resource allocation. Therefore, improvement in this parameter leads to a reduction in delays for users, enhancing their satisfaction. Since the proposed solution utilizes the GWO algorithm and creates optimal load balance in the fog computing infrastructure, the average execution time for host selection in BCOFF has decreased by over 20%, significantly impacting the quality-of-service improvement. In Figure 8, the energy consumption in the solutions is examined.

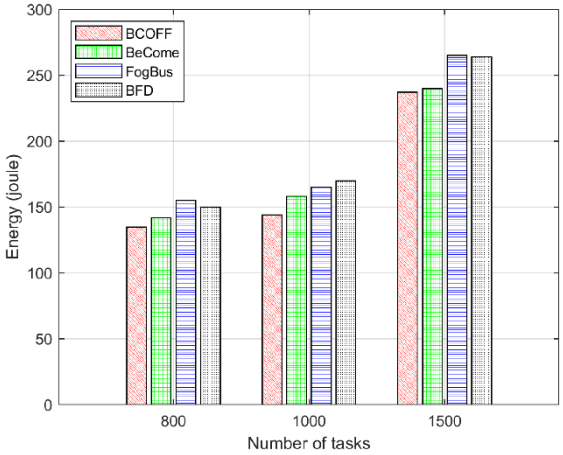

Figure 8. The amount of energy consumption of the evaluated solutions (joules).

Given that the core algorithm of BCOFF utilizes the GWO algorithm with the capability of a resource examiner, it obtains relevant information about the status of nodes. Indeed, idle nodes can be identified by employing this capability, and more energy savings can be achieved by halting them. Additionally, this capability results in offloading among the nodes, potentially preventing an increase in energy consumption. As seen in Figure 8, an average energy savings of over 10% has been achieved in each of the three evaluations.

To clarify this seemingly counterintuitive result, the overall energy consumption in BCOFF can be decomposed into three main components: (i) computation energy at fog nodes, (ii) communication energy during task offloading and result delivery, and (iii) consensus-related energy overhead from blockchain operations. While the VNPoS-based validation mechanism introduces

additional communication and lightweight cryptographic operations, its energy overhead remains limited due to the absence of intensive mining and the use of a validator-selection heuristic tailored for fog environments. More importantly, GWO-based resource-aware offloading reduces unnecessary task migrations, node congestion, and frequent server shutdowns. These effects lead to substantial savings in computation energy, which outweigh the modest consensus overhead. Consequently, the net energy consumption of the BCOFF framework is lower than that of baseline approaches, despite the blockchain layer. This explains why the proposed solution achieves overall energy savings while maintaining integrity guarantees.

This is attributed to the effective utilization of the GWO algorithm and improved resource allocation, which significantly improves energy efficiency compared to other solutions, including FogBus and BFD. Compared to BeCome, which utilizes a resource allocation based on NSGAII in its core, BCOFF shows greater energy efficiency. On the other hand, since BCOFF employs the VNPoS algorithm for transaction validation in its core, it results in reduced transaction execution time, positively impacting the energy consumption of the solution. This has led to greater efficiency compared to other solutions. In Figure 9, the number of times servers have been shut down is displayed. This capability, if not performed correctly, can hurt the processing nodes.

mentioned earlier, if a server is not correctly halted, turning it on again can increase both energy consumption and execution time. Examining the execution time related to a solution like BFD also indicates the same reality. While BFD may have turned off more machines through its algorithm, as previously discussed, both the execution time and energy consumption of BFD are higher than those of BCOFF.

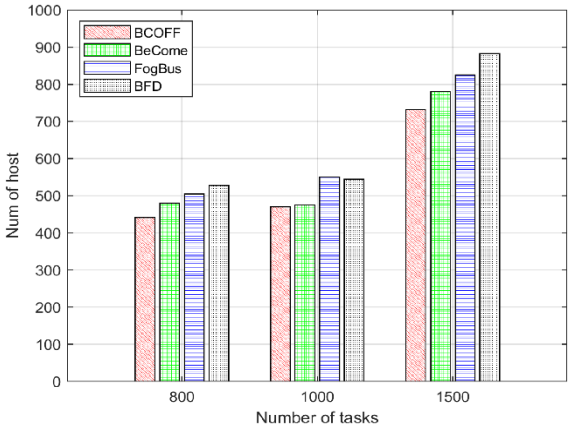

Figure 9. The number of times the processing nodes shut down

The shutdown nodes metric represents the number of times a fog node is transitioned from an active to an idle (powered-off) state during the simulation. Node shutdown decisions are governed by a resource-aware policy, in which a node is powered off only when it remains idle for a predefined interval and no pending tasks are assigned to it.

Table 5. Numerical results of the evaluation.

Metric	BCOFF			BeCome			FogBus			BFD		
Number of tasks	800	1000	1500	800	1000	1500	800	1000	1500	800	1000	1500
execution time	82	95	300	112	120	355	115	129	335	126	140	435
Host selection	7	6	9	9	8	10.5	9	8.5	11	10	9.5	12
Energy consumption	135	144	237	142	158	240	155	165	265	150	170	264
Shutdown nodes	441	470	732	475	480	780	505	550	825	528	545	885
Delay	0.6	1	2.5	0.67	1	3.35	0.62	0.95	5.2	0.88	1.2	4

If a node is mistakenly turned off, it not only negatively affects the load balance and execution time but also requires the node to be powered on again if needed, leading to increased energy consumption. As seen in Figure 9, although in the proposed solution the number of turned-off nodes has been generally lower, considering the energy consumption parameter discussed in Figure 8, it can be inferred that in BCOFF, server shutdowns are performed based on the status of available resources and nodes, and servers are turned off only when not needed. This is crucial because, as

This prevents unnecessary shutdowns that could negatively affect load balance or execution time. Transition overheads associated with powering nodes on and off are not explicitly modeled in iFogSim and are therefore not included in the energy calculations. However, since shutdowns are performed only for persistently idle nodes, the impact of transition costs is assumed to be negligible relative to the overall energy savings. This assumption is consistent across all compared methods, ensuring a fair comparison.

In Figure 10, the solutions are evaluated based on the delay parameter. One of the goals of employing fog computing is to reduce latency, so examining this parameter can also be insightful. As observed, BCOFF performs better than other solutions in this evaluation. While the delay of the proposed solution is slightly less than FogBus only in the first evaluation, in the third evaluation, as the number of tasks increases, BCOFF proves to be much more efficient than the FogBus algorithm. In other words, as the number of requests increases, FogBus loses its efficiency to a significant extent, but BCOFF continues to perform optimally. Therefore, deploying BCOFF in scalable environments may be more favorable.

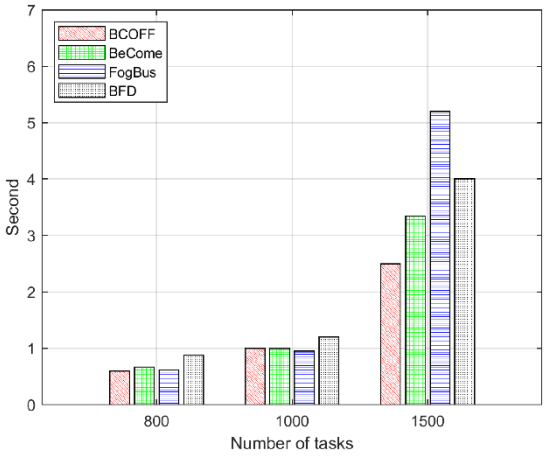

Figure 10. The amount of delay in the evaluated solutions.

Table 5 shows the numerical results from the evaluation of the solutions, and Figure 10 shows a comparable diagram. A dedicated evaluation of consensus-layer throughput and latency is left for future work, as the current study focuses on end-to-end performance in fog offloading scenarios.

5. Conclusion

Fog computing is a distributed processing architecture involving many heterogeneous devices connected to the network in the Fog to provide pervasive services such as processing, network communication, and storage. One prominent feature of this type of processing is the expansion of cloud services to network edges. This enables processing resources, network communications, control, and storage to be closer to the end user, reducing the time and network overhead for data transmission. As processing resources in Fog computing are close to mobile devices, complex processing tasks on mobile devices are offloaded to nodes at the network edge, allowing access to high-processing resources with low latency. Considering the impact of offloading on service delivery and user satisfaction, this article employs the GWO algorithm to improve the offloading process. By leveraging the algorithm's ability to assess available resources, it selects the most suitable nodes based on resource availability. Under the adopted execution-time and power-based energy model, this approach reduces execution time and enhances overall energy efficiency.

However, various security and data integrity issues may arise during the transfer process due to leakage or flaws in transfer operations. When a processing task is sent to a node at the edge using the offloading algorithm, the node may not be able to distinguish normal users from attackers, potentially violating privacy and compromising the security of transferred data. This article employs a Blockchain-based Verifiable Non-Repudiation Offloading Strategy (VNPoS) to address these concerns. After determining the offloading strategy, information related to offloading is evaluated by decentralized processing centers rather than a central controller. Each server's information can be encapsulated into a block and utilized within the blockchain to record cryptographic commitments (hashes) of offloading events, while detailed task payloads and user-related data remain off-chain to limit information exposure.

Except for the machine providing the desired service to the mobile device, other machines compete to validate the record related to the offloading operation for verification. Once all machines have reviewed the service-related information, it can be ensured that security and integrity have been fully enforced during the offloading process, and the offloading operation begins.

The results of the proposed solution indicate an improvement in service quality. Within the adopted simulation assumptions and the standard power-based energy abstraction used in iFogSim, the proposed solution improved end-to-end service quality by reducing execution time, lowering modeled energy consumption, and decreasing delay. Despite these promising results, this study has several limitations. The proposed VNPoS mechanism is designed as a lightweight validator-selection heuristic and does not provide formal cryptographic security guarantees. A comprehensive adversarial threat analysis and formal security proofs are beyond the scope of this work.

Additionally, although blockchain integration enhances integrity, the evaluation does not include a dedicated micro-benchmark of consensus-layer overhead under diverse network and workload

conditions. Furthermore, the current energy model follows a simplified power-times-time abstraction and does not explicitly capture fine-grained utilization dynamics, thermal effects, or hardware-level power states; extending the framework with more physically detailed energy formulations remains future work.
Addressing these limitations through formal security modeling and extended consensus-level evaluation is identified as an important direction for future research.

6. Future Works

Despite the successful simulation and evaluation of the BCOFF algorithm, this research still needs to be exhaustive. Future work can focus on several areas:

- Handling larger datasets: The current evaluation considers task workloads ranging from 800 to 1500 tasks within a fog computing environment. Future work will extend this analysis to significantly larger task volumes and fog-node scales to further investigate robustness and scalability under stress conditions.
- Security analysis: Although the study has discussed the potential benefits of the BCOFF algorithm, an in-depth security analysis could be beneficial.
- Improving VNPoS algorithm: Based on the insights gained from this simulation, further enhancements could be made to the VNPoS algorithm to improve its performance.

Declarations

- Ethics approval and consent to participate: This is not applicable.
- Consent for publication: This is not applicable.
- Availability of data and materials: The data supporting the findings of this study are available from the corresponding author upon reasonable request.
- Competing interests: The authors declare that they have no competing interests.
- Funding: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.
- Authors' contributions: Sajjad Daliri and Somayyeh Jafarali Jassbi contributed to the conception, design, methodology, writing, and final approval of the manuscript.
- Acknowledgments: The authors would like to thank the Department of Computer Engineering, Science and Research Branch, Islamic Azad University, for their support.

References

[1] H. K. Apat, R. Nayak and B. Sahoo, "A comprehensive review on Internet of Things application placement in Fog computing environment," *Internet of Things, 100866,* vol. 23, no. 4, October 2023.

[2] M. Sheikh Sofla, M. Haghi Kashani, E. Mahdipour and R. Faghih Mirzaei, "Towards effective offloading mechanisms in fog computing," Multimedia Tools and Applications, vol. 81, no. 2, pp. 1997-2042, 2022.

[3] W. Wang, J. Chen, Y. Jiao, J. Kang, W. Dai and Y. Xu, "Connectivity-Aware Contract for Incentivizing IoT Devices in Complex Wireless Blockchain," *IEEE Internet of Things Journal,* vol. 10, no. 12, pp. 10413 - 10425, 2023.

[4] R. Das and M. M. Inuwa, "A review on fog computing: Issues, characteristics, challenges, and potential applications," *Telematics and Informatics Reports,* vol. 10, 2023.

[5] A. Hazra, M. Adhikari, T. Amgoth and S. N. Srirama, "Fog Computing for Energy-Efficient Data Offloading of IoT Applications in Industrial Sensor Networks," *IEEE Sensors Journal,* vol. 22, no. 9, pp. 8663-8671, 2022.

[6] M. M. Razaq, B. Tak, L. Peng and M. Guizani, "Privacy-Aware Collaborative Task Offloading in Fog Computing," *IEEE Transactions on Computational Social Systems,* vol. 9, no. 1, pp. 88-96, 2022.

[7] D. Wanchun, W. Tang, B. Liu, X. Xu and Q. Ni, "Blockchain-based Mobility-aware Offloading mechanism for Fog computing services," *The International Journal for the Computer and Telecommunications Industry,* vol. 164, pp. 261-273, 2020.

[8] M. Kiani and M. R. Khayyambashi, "An Evaluation of New Meta-heuristics for Virtual Machine Placement in Cloud Data Centers," *Journal of AI and Data Mining (JAIDM),* vol. 13, no. 1, pp. 1-10, 2025.

[9] A. Ometov, O. L. Molua, M. Komarov and J. Nurmi, "A Survey of Security in Cloud, Edge, and Fog Computing," *Sensors*, vol. 22, no. 3, pp. 927-953, 2022.

[10] M. Orabi, R. Al Barghash and S. Abbas, "Dynamic Offloading in Fog Computing: A Survey," in *Proceedings of International Conference on Information Technology and Applications, Dubai, UAE*, 2022.

[11] M. A. Alizadeh, S. J. Jassbi, A. Khademzadeh and M. Haghparast, "Novel lightweight and fine-grained fast access control using RNS properties in fog computing," *Cluster Computing*, vol. 27, no. June 2024, pp. 3799-3817, 2023.

[12] N. Kumari, A. Yadav and P. K. Jana, "Task offloading in fog computing: A survey of algorithms and optimization techniques," *Computer Networks*, vol. 214, no. 7, pp. 109-137, 2022.

[13] B. Kar, Y. Widhi, L. Ying-Dar and A. Asad, "Offloading Using Traditional Optimization and

Machine Learning in Federated Cloud–Edge–Fog Systems: A Survey," IEEE Communications Surveys & Tutorials, vol. 25, no. 2, pp. 1199-1226, 2023.

[14] K. Gasmi, S. Dilek, S. Tosun and S. Ozdemir, "A survey on computation offloading and service placement in fog computing-based IoT," *The Journal of Supercomputing*, vol. 78, pp. 1983-2014, 2022.

[15] S. Khezr, A. Yassine and R. Benlam, "Towards a secure and dependable IoT data monetization using blockchain and fog computing," *Cluster Computing*, vol. 26, no. 2, pp. 1551-1564, 2023.

[16] Y. I. Alzoubi, A. Al-Ahmad and H. Kahtan, "Blockchain technology as a Fog computing security and privacy solution: An overview," *Computer Communications*, vol. 182, pp. 129-152, 2022.

[17] D. Wu and N. Ansari, "A Cooperative Computing Strategy for Blockchain-Secured Fog Computing," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6603 - 6609, 2020.

[18] S. Guo, X. Hu, S. Guo, X. Qiu and F. Qi, "Blockchain Meets Edge Computing: A Distributed and Trusted Authentication System," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1972 - 1983, 2020.

[19] F. Jameel, M. A. Javed, S. Zeadally and R. Jäntti, "Efficient Mining Cluster Selection for Blockchain-Based Cellular V2X Communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4064 - 4072, 2021.

[20] M. Madine, K. Salah, R. Jayaraman, Y. Al-Hammadi, J. Arshad and I. Yaqoob, "appXchain: Application-Level Interoperability for Blockchain Networks," *IEEE Access*, vol. 9, no. 1, pp. 87777-87791, 15 6 2021.

[21] N. C. Gowda, S. S. Manvi, B. A. Malakreddy and P. Lorenz, "BSKM-FC: Blockchain-based secured key management in a fog computing environment," *Future Generation Computer Systems*, vol. 142, pp. 276-291, 2023.

[22] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi and W. Dou, "BeCome: Blockchain-Enabled Computation Offloading for IoT in Mobile Edge Computing," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4187 - 4195, 2020.

[23] J. Cui, F. Ouyang, Z. Ying, L. Wei and H. Zhong, "Secure and Efficient Data Sharing Among Vehicles Based on Consortium Blockchain," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8857 - 8867, 2022.

[24] N. Sharma and R. Rohilla, "Scalable and Cost-Efficient PoA Consensus-Based Blockchain Solution for Vaccination Record Management," *Wireless Personal Communications*, vol. 135, pp. 1177-1207, 07 May 2024.

[25] J. Sun, Y. Zhang and M. Trik, "PBPHS: A Profile-Based Predictive Handover Strategy for 5G Networks," *Cybernetics and Systems*, vol. 55, no. 5, pp. 1041-1062, 2022.

[26] N. Premkumar and R. Santhosh, "Pelican optimization algorithm with blockchain for secure load balancing in fog computing," Multimedia Tools and Applications, vol. 83, pp. 53417-53439, 2023.

[27] W. Dou, W. Tang, B. Liu, X. Xu and Q. Ni, "Blockchain-based Mobility-aware Offloading mechanism for Fog computing services," *Computer Communications*, vol. 164, pp. 261-273, 2020.

[28] P. Lang, D. Tian, X. Duan, J. Zhou, Z. Sheng and V. C. M. Leung, "Blockchain-Based Cooperative Computation Offloading and Secure Handover in Vehicular Edge Computing Networks," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 7, pp. 3839 - 3853, 2023.

[29] H. Liao, Y. Mu, Z. Zhou, M. Sun, Z. Wang and C. Pan, "Blockchain and Learning-Based Secure and Intelligent Task Offloading for Vehicular Fog Computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4051 - 4063, 2021.

[30] M. Salimian, M. Ghobaei-Arani and A. Shahidinejad, "Toward an autonomic approach for Internet of Things service placement using gray wolf optimization in the fog computing environment," *Software: Practice and Experience*, vol. 51, no. 8, pp. 1745-1772, 2021.

[31] G. Wang, J. Wu and M. Trik, "A Novel Approach to Reduce Video Traffic Based on Understanding User Demand and D2D Communication in 5G Networks," *IETE Journal of Research*, pp. 1-17, 2023.

[32] J. Yadav and S. Sangwan, "E-MOGWO Algorithm for Computation Offloading in Fog Computing," *Intelligent Automation & Soft Computing*, vol. 36, no. 1, pp. 1063-1078, 2022.

[33] E. Khezri, R. O. Yahya, H. Hassanzadeh, M. Mohaidat, S. Ahmadi and M. Trik, "DLJSF: Data-Locality Aware Job Scheduling IoT tasks in fog-cloud computing environments," *Results in Engineering*, vol. 21, 2024.

[34] J. Singh, P. Singh and S. Singh, "Fog computing: A taxonomy, systematic review, current trends and research challenges," *Journal of Parallel and Distributed Computing*, vol. 157, pp. 56-85, 2021.

[35] E. Khezri, R. O. Yahya, H. Hassanzadeh, M. Mohaidat, S. Ahmadi and M. Trik, "DLJSF: Data-Locality Aware Job Scheduling IoT tasks in fog-cloud computing environments," *Results in Engineering*, vol. 21, 2024.

[36] R. Peres, M. Schreier, D. A. Schweidel and A. Sorescu, "Blockchain meets marketing: Opportunities, threats, and avenues for future research," *International Journal of Research in Marketing*, vol. 40, no. 1, pp. 1-11, 2023.

[37] M. Trik, A. M. Norouzzadeh Gil Molk, F. Ghasemi and P. Pouryeganeh, "A Hybrid Selection Strategy Based on Traffic Analysis for Improving Performance in Networks on Chip," *Journal of Sensors*, vol. 2022, no. Special, 2022.

[38] F. A. Saif, R. Latip, Z. M. Hanapi and K. Shafinah, "Multi-Objective Grey Wolf Optimizer Algorithm for Task Scheduling in Cloud-Fog Computing," *IEEE Access*, vol. 11, pp. 20635 - 20646, 2023.

[39] M. Wendl, M. H. Doan and R. Sassen, "The environmental impact of cryptocurrencies using proof of work and proof of stake consensus algorithms: A systematic review," *Journal of Environmental Management*, vol. 326, no. A, 2023.

[40] R. W. Ahmad, K. Salah, R. Jayaraman, I. Yaqoob, S. Ellahham and M. Omar, "Blockchain and COVID-19 pandemic: applications and challenges," *Cluster Computing*, vol. 26, no. June 2024, p. 2383–2408, 2023.

[41] T. Hardjono, A. Lipton and A. Pentland, "Toward an Interoperability Architecture for Blockchain Autonomous Systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1298 - 1309, 2019.

[42] Y. Pang, "A New Consensus Protocol for Blockchain Interoperability Architecture," *IEEE Access*, vol. 8, no. 18, August, pp. 153719 - 153730, 2020.

[43] S. Tuli, R. Mahmud, S. Tuli and R. Buyya, "FogBus: A Blockchain-based Lightweight Framework for Edge and Fog Computing," *Journal of Systems and Software*, vol. 265, no. August 2019, pp. 22-36, 2019.

[44] T. Tlili and S. Krichen, "Best Fit Decreasing Algorithm for Virtual Machine Placement Modeled as a Bin Packing Problem," in *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT), Rome*, 2023.

BCOFF: چارچوب مبتنی بر زنجیره‌بلوکی با پروتکل اجماع برای افزایش کارایی و تضمین یکپارچگی واگذاری پردازش در محیط‌های پردازش مه

سجاد دلیری و سمیه جعفرعلی جاسبی*

دانشگاه آزاد اسلامی، واحد علوم و تحقیقات، دانشکده مهندسی کامپیوتر، تهران، ایران.

ارسال ۲۰۲۵/۰۷/۲۷؛ بازنگری ۲۰۲۶/۰۶/۰۵؛ پذیرش ۲۰۲۶/۰۶/۰۶

چکیده:

رشد سریع اینترنت اشیا (IoT) چالش‌های متعددی را در محیط‌های پردازش مه از جمله تأخیر سرویس، محدودیت منابع و مدیریت اعتماد ایجاد کرده است. پردازش مه با انتقال قابلیت‌های پردازشی و ذخیره‌سازی به نزدیکی دستگاه‌های نهایی، بخشی از این محدودیت‌ها را برطرف می‌کند. با این حال، فرایند واگذاری پردازش همچنان با مشکلاتی نظیر انتخاب بهینه منابع، تضمین یکپارچگی داده‌ها و قابلیت اعتماد مواجه است. در این پژوهش، چارچوب BCOFF را به‌عنوان یک راهکار امن و کارآمد ارائه شده است که به‌صورت همزمان تخصیص منابع را بهینه‌سازی کرده و امکان واگذاری پردازش قابل اعتبارسنجی را فراهم می‌کند. در چارچوب پیشنهادی، الگوریتم بهینه‌سازی گرگ خاکستری (GWO) برای انتخاب بهینه منابع و تخصیص وظایف به کار گرفته شده و زنجیره بلوکی به‌عنوان یک دفترکل توزیع‌شده مقاوم در برابر دستکاری، رویدادهای مرتبط با واگذاری پردازش را ثبت و اعتبارسنجی می‌کند. همچنین سازوکار اجماع سبک وزن VNPoS که اقتباسی از مدل NPoS است، برای انتخاب اعتبارسنج‌ها و کاهش سربار اجماع در محیط‌های ناهمگون پردازش مه ارائه شده است. نتایج ارزیابی در محیط شبیه‌سازی iFogSim و برای بارهای کاری شامل ۸۰۰ تا ۱۵۰۰ وظیفه نشان می‌دهد که BCOFF زمان اجرا را بین ۱۵ تا ۲۷ درصد، تأخیر انتخاب میزبان را بین ۲۲ تا ۲۵ درصد و مصرف انرژی را بین ۵ تا ۹ درصد نسبت به روش‌های موجود کاهش می‌دهد. نتایج حاصل نشان می‌دهد که ترکیب زمانبندی مبتنی بر GWO با سازوکار زنجیره بلوکی VNPoS می‌تواند چارچوبی کارآمد، مقیاس‌پذیر و قابل اعتماد برای واگذاری پردازش در محیط‌های پردازش مه فراهم سازد.

کلمات کلیدی: پردازش مه، اینترنت اشیا، واگذاری پردازش، زنجیره بلوکی، پروتکل اجماع.