EasyChair Preprint

№ 11430

Blockchain Reinforced Task Distribution and Secure Deduplication Using Adaptive Deep Reinforcement Learning in Cluster Based Fog IoT

Boniface Ntambara and Alexandre Niyomugaba

December 1, 2023

Blockchain Reinforced Task Distribution and Secure Deduplication using Adaptive Deep Reinforcement Learning in Cluster based Fog IoT

Boniface NTAMBARA
Department of IoT-Embedded and Mobile Systems,
The Nelson Mandela African Institution of Science and Technology
P.O BOX 447 Arusha-Tanzania
bonifacen@ieee.org /bonifacen@nm-aist.ac.tz

ORCID: https://orcid.org/my-orcid?orcid=0000-0001-9915-5421

Alexandre Niyomugaba
Department of Embedded and Mobile Systems
The Nelson Mandela African Institution of Science and Technology
P.O BOX 447 Arusha-Tanzania

niyomugabaa@nm-aist.ac.tz
ORCID: 0000-0002-8133-5253

***Abstract*— The fog-assisted Internet-of-Things (IoT) is gaining interest due to its large number of devices, which can lead to more duplicate data transmission over the internet. This paper proposes task distribution and secure deduplication over Cluster-based IoT, implementing four layers: IoT Devices Layer, Fog Layer, Cloud Layer, and Service Layer. In the IoT devices layer, devices sense air pollutants and are authenticated to the cloud server using Edwards Curve-based Elliptic Curve Cryptography (EC-ECC). Adaptive Rewards Optimized Deep Reinforcement Learning (ARO-DRL) is used for cluster-head selection at the first layer. In the fog layer, SHA-3 is proposed for duplicate verification, and the Emperor Penguin Optimization Algorithm is used to choose the best fog node. Packet Scrutinization Algorithm is used in the fog node to analyze packet features, including DDoS attack packets. A proxy server is deployed between the cloud server and fog node for queue modeling. In the cloud layer, a hybrid cloud environment is used to protect organizations' data in a highly secure manner. IoT devices are divided into sensitive and nonsensitive devices, with sensitive data encrypted using RC6, AES, and Fiestel encryption schemes. The overall environment is assumed to be decentralized, with security invoked to IoT devices to provision Quality of Service (QoS) by avoiding attackers. Experiments were conducted and analyzed using NS3 with Java programming, and simulation results showed improvements in average latency, user satisfaction, network lifetime, energy consumption, and security strength.**

Keywords*— Fog Assisted Internet of Things, Task Allocation, Secure Deduplication, Secure Clustering, and NS3 with Java, and Blockchain.*

I. INTRODUCTION

The Industrial Sector is a prime application area for Internet of Things (IoT) technologies, particularly wireless sensor actuator networks (WSAN) and wireless sensor networks (WSN). These technologies aid in sensitive information management, such as energy efficiency, air quality management, fault prediction, resource prediction, and product planning. Key WSN-IoT applications include smart city, smart home, smart transportation, disaster management, smart grids, energy control systems, smart healthcare, urban terrain tracking, smart agriculture, and industrial IoT. However, preserving energy efficiency without affecting communication among IoT entities remains a challenge due to the involvement of numerous nodes, communication among multiple entities, multi-hop communication, dynamic topology of the network, and lack of optimized network design. Despite these challenges, diverse research has been conducted to manage the massive growth of IoT devices and sensors.

This research paper addresses the challenges in Wireless Sensor Networks (WSN-IoT) and proposes novel methodologies to achieve energy efficiency in an IoT-Fog-Cloud connected environment. Factors affecting energy consumption and network lifetime include idle listening, node isolation, data transmission, overhearing, redundant data, collisions, and frequent retransmissions. Security is a challenging task in the IoT environment, and integrating IoT with the cloud offers storage of security credentials. IoT devices are designed with low power and resources, making them efficient and suitable for integrating with a cloud platform [1].

IoT-enabled cloud provides ubiquitous computing for easier and faster access, and this integration is studied in the four-tier architecture of IoT devices, network devices, edge computing, and cloud layer. The elasticity of the cloud has enabled integration for multiple applications [2]. IoT aims to provide efficient communication for connected devices, but proper maintenance is necessary to minimize attacker participation. Conventional algorithms and mechanisms were presented to resolve security issues in the developed system. Fig. 1 depicts the IoT-enabled cloud environment, which is also subject to attacker involvement, as attackers can be of any type and their goals may vary. Conventional algorithms and mechanisms were presented to resolve security issues and protect the system against attackers [3].

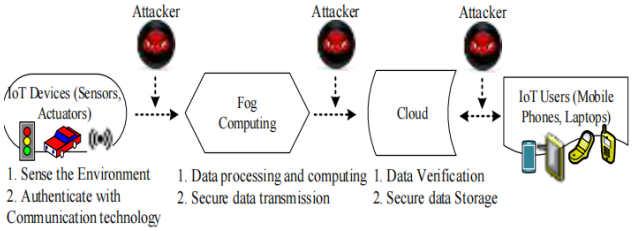

Fig. 1. IoT-integrated Fog Cloud.

The integration of IoT environments requires security measures. Fog computing is an emerging paradigm that uses

large fog nodes to reduce user delays and send remaining traffic to cloud data centers. Optimization methods like weighted sum, hierarchical, and trade-off methods have been proposed to optimize this model, aiming to reduce processing time and improve user experience [4]. Fog computing offers heterogeneity, online analytics, large-scale IoT applications support, and easy cloud interplay [5]. It is particularly useful in healthcare, air pollution monitoring, smart grids, smart homes, and smart vehicles. Air pollution, caused by harmful gases, dust/fumes, and odors, affects human life, animals, and plants [6]. Fog computing can help mitigate these issues by detecting and addressing pollutants like nitrogen dioxide, carbon dioxide, carbon monoxide, methane, hydrogen sulphide, hydrocarbons, and ozone [7]. Air pollution monitoring systems often face a significant amount of duplicate data, which can increase storage capacity and efficiency. To address this, a data deduplication scheme is needed to eliminate redundant or similar data. IoT devices submit this data to cloud servers for storage [8-9]. Fog computing is used for task allocation, reducing latency, communication overhead, and communication cost. Clustering is an important process in fog-enabled WSNs, with Cluster Head (CH) selected on each cluster and other nodes in the cluster members (CMs) [10-11]. CH functions by controlling random selection, exploiting heterogeneity energy thresholds to avoid residual energy nodes, optimizing the minimum distance between CHs and fog nodes [12]. Fog computing increases energy efficiency and reduces overhead by aggregating sensing data and forwarding computing results to cloud servers. It is crucial for task allocation and low latency in real-world environments. Data confidentiality is also important, as large amounts of data can cause attacks. Cloud servers extract this information and provide on-demand services to end-users, but they do not guarantee data security and integrity [13-14]. The fog-assisted cloud environment for IIoT applications uses blockchain architecture, which includes blocks with transaction lists, hash values, and timestamps [15]. The ProvChain scheme is introduced to ensure tamper-proof records. A privacy-preserving model for secure data storage is proposed using blockchain, and blockchain-based forensic architecture is used for vehicular network environments to analyze accident cases [16]. However, cloud servers outsource non-sensitive information to end-users, which does not guarantee data security and integrity as shown in Fig. 2.

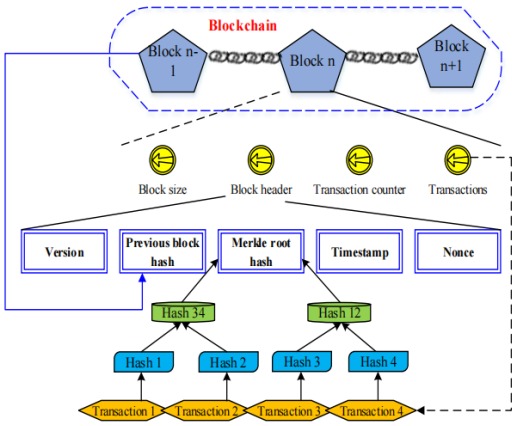

Fig. 2. Blockchain structure.

This paper presents a new innovation in remote monitoring and control system design for applications, combining Industrial IoT, fog nodes, and cloud environments. The paper addresses issues in integrating these technologies, focusing on task allocation and secure deduplication. IoT devices serve both sensing and actuation functions, while cloud systems store historical information and enable remote monitoring of sensing environments. Fog nodes act as faster gateways in Industry 4.0.

1.1 Motivation & Contributions

This paper discusses the challenges in fog-enabled IoT, focusing on the large latency issues faced by devices due to cloud server issues. IoT devices are designed for various applications, including connecting multiple sensors and actuators, making real-time data decisions, monitoring and reporting status, and analyzing large-scale data. However, security is a common concern in IoT, especially when transferring sensitive data. The paper highlights the need for improved security measures in IoT to address these challenges.

This paper proposes three key solutions for efficient authentication of IoT devices: multi-factor authentication, lightweight algorithms, blockchain technology, energy-efficient clustering, and scheduling tasks via queue management. These solutions aim to reduce resource consumption and improve the system's effectiveness. The main contributions include registering all IoT devices to a cloud server using Edwards Curve Elliptic Curve Cryptography (EC-ECC), clustering similar devices using Adaptive Rewards Optimized Deep Reinforcement Learning algorithm, implementing secure data deduplication using SHA-3, and selecting the optimal fog node using Emperor Penguin Optimization Algorithm (EPO). A proxy server is deployed in the fog layer, scheduling packets into real-time and non-real-time classes using the M/M/C model. Data packets are encrypted using lightweight encryption algorithms such as RC6, Fiestel, and AEs before transmission to the cloud server. The experimental results show that the proposed scheme outperforms previous works based on QoS metrics such as average latency, energy consumption, user satisfaction, network lifetime, and security strength. The proposed scheme is expected to reduce computations in the system for device authentication.

Table 1 describes the notations and descriptions of the symbols and acronyms in the paper.

TABLE 1. NOTATIONS AND DESCRIPTIONS.

Notation	Description
d_n	IoT devices in the system $n = 1,2,3, \ldots \ldots$
ID_n	Identity of each IoT device $n = 1,2,3, \ldots \ldots$
d_t	Type of IoT device
R	Random number
sk_n	Secret key of each IoT device $n = 1,2,3, \ldots \ldots$
$L_{n(x,y)}$	Location of each IoT device $n = 1,2,3, \ldots \ldots$
Pid	Generated new identity
$E_{KF}(B)$	Encrypted PUF biometric
Pk_n	SRAM-PUF based public key of IoT device
Pr_n	SRAM-PUF private key IoT device
$S_i(D_n)$	Signature with data structure of n^{th} device
$b_{id}(d_n)$	Block identity of n^{th} device
$T_b(d_n)$	Block timestamp of n^{th} device

1.2 Paper Organization

The paper presents a literature review on fog assisted IIoT, presents major problem statements, briefly explains the proposed system design and architecture, and presents each new idea in an organized manner. Experimental settings for the proposed system design are presented, and comparisons between the proposed and previous approaches are evaluated. The paper concludes with a summary of the paper's conclusion and future work.

II. REVIEW OF RELATED WORKS

2.1 Clustering in IoT / WSN

A hyper round policy-based clustering scheme has been proposed to improve network lifetime by controlling frequent re-clustering using a fuzzy inference system [18]. Random nodes are chosen, and cluster formation follows TDMA slots. The re-clustering process is triggered by the fuzzy inference system to improve network lifetime. However, frequent clustering and reclustering increase energy consumption due to frequent control packet exchange. A fuzzy power optimized clustering algorithm was proposed to reduce energy consumption in Wireless Sensor Networks (WSN) [19]. The optimal clustering is selected based on multi-parameter iteration, considering factors like centralism and distance with the base station. This approach relies heavily on node density, which is not suitable for energy-aware networks. In REECHD, the CH selection is performed based on probability value as follows in equation (1).

$$CH_{prob} = \max\left(\frac{C_{prob}}{K}\left(\frac{E_{Residual}}{E_{max}} + IW^{-1}\right), P_{min}\right) \quad (1)$$

Here the leader election probability ($CHprob$) is computed in terms of predefined initial probability ($Cprob$), minimum probability value a CH must have ($Pmin$), residual energy ($EResidual$) and the constant value (K).

The whale optimization algorithm, which is self-adaptive, has been implemented for optimal CH selection in large-scale networks, but its effectiveness may not be suitable for distant nodes [20]. The authors analyze SAWOA's self-adaptive whale optimization algorithm using benchmark optimization techniques. They propose an optimal clustering algorithm for lifetime maximization (LiMCA) in WSN-IoT, balancing energy consumption among whales and forming clusters. However, non-optimal CH selection increases the number of rotations and re-clustering. Numerous optimization techniques focus on cluster formation.

2.2 Task Allocation & Security Schemes

A multi-criteria-based decision-making approach for task allocation in several nodes, implemented at edge nodes or presented in peer topology. The scheme follows two decisions for optimal task allocation, addressing high energy consumption due to high latency. Spatial crowdsourcing assisted task owners-based task allocation and data aggregation are proposed through fog computing, which allows servers to collect sensed information from mobile users and distribute and aggregate data in a privacy-aware manner [21]. IIoT-based fog computing technology is presented for smart factory applications, with hierarchical fog servers-based deployments categorizing sensed data into high priority and low priority. High priority requests are scheduled first due to emergency/urgent demands, and a workload assignment algorithm is used to offload high traffic load to higher fog tiers. However, end-to-end delay is large due to the large number of workloads at fog tiers [22]. Adaptive configuration of fog nodes over IIoT environments offers IoT services such as imminent failure detection and automatic monitoring control, improving industrial system performance [23]. Lyapunov optimization and parallel Gibbs sampling methods are proposed for adaptive fog node configuration, but not adopted for real-time applications [24]. Smart resources partitioning is proposed in fog assisted IIoT environments, using Zipfs law to compute the relationship between popularity ranks of computing control layer and data processing layer [25]. Hybrid approaches like reinforcement learning and fuzzy logic algorithms are proposed to minimize latency for healthcare applications over IoT environments, but their limitations include high service latency in the application layer [26-27]. Security is also essential for fog-enabled cloud environments, with matrix-based key agreement and lightweight authentication models being proposed for communication and verifying multiple party identities. The Intelligent Transportation Control System (FSF-ITLCS) framework addresses various security attacks, but overall computation time is high [28].

2.3 Smart Applications in IoT

Air pollution is a significant concern that affects the health of humans, animals, and plants. It has various applications in areas such as roadside pollution monitoring, industrial perimeter monitoring, site selection for reference monitoring stations, and indoor air quality monitoring [29]. In recent years, there has been a focus on air pollution monitoring systems in Wireless Sensor Networks (WSN), which use data from sensors to analyze and compute air quality indexes (AQIs) to visualize air quality locations efficiently [30]. One such system is the IoT-based 3D air quality sensing system, which is designed as a real-time, power-efficient, and fine-grained architecture with four layers: sensing layer (data collection), transmission layer (bidirectional communications support), processing layer (data processing and analysis), and presentation layer (provide graphical interface for users) [31-32]. However, data security is not considered in these systems [33]. To protect the air quality monitoring system framework, authors have focused on data integrity and security for low-cost air quality sensors used to collect sensor

information and manage pollutants under three cases: Sensor in Physical Possession, Sensor MAC address knowing (geographical information) environment, and Automatic air pollution monitoring in large-scale environments [34]. Fog computing has been proposed to tackle mobile crowdsensing challenges by ensuring data confidentiality and task allocation based on user mobility. Fog nodes detect and remove replicate data using BLS-obvious pseudorandom number function and chameleon hash function, which hide users' information to anonymous mobile users. However, this approach may not be suitable for applications requiring large volumes of data, as task allocation/assignment is not effective, and the process is time-consuming [35]. A new P-SEP based fog computing model was proposed, which reduces energy usage and increases network lifetime. However, the clustering process is not effective, as it randomly selects adjacent fog nodes without properly investigating and implementing optimal fog node allocation [36]. Adaptive block compressed sensing was proposed, which is based on sensor-cloud data acquisition methods over fog environments but has drawbacks such as being large complex and not lightweight [37]. Additionally, it causes high energy utilization in fog nodes due to virtual cluster formation in the lower WSN layer. In IIoT, fog-enabled cloud environments are considered, but IoT devices are vulnerable and insecure to several threats [38]. Secure KNN was proposed to ensure data confidentiality, but it is expensive and has high data searching time. Furthermore, KNN is not suitable for dense areas or processing large amounts of data, particularly in real-world dataset processing. The proposed scheme addresses these issues and improves air pollution monitoring systems [39].

III. PROBLEM FORMULATION

Fog computing faces challenges in secure task management due to the consumption of resources and performance degradation caused by processing unauthorized user tasks. Task scheduling and queue management in fog environments are based on limited parameters and conventional FIFO policy, leading to large waiting times. Task offloading is handled using either task or fog-oriented metrics, but both are necessary for better efficiency. The particle Swarm Optimization (PSO) algorithm was proposed to enhance energy efficiency, but it leads to higher time consumption for routing and clustering sensor nodes in the network. Hierarchical data fusion methods for smart healthcare consider biosensor readings for patient health status monitoring, but environmental factors also play a pivotal role. CPE-based analysis is not suitable for real-time analysis due to user differences and parameters. Task scheduling and offloading are performed based on task priority, but offloading decisions by gateways increase time and complexity. Queues follow FIFS policy, increasing waiting time and slack time. Priority values in the LP queue are given high priority, but increasing priority affects computational time for HP tasks. Energy-efficient offloading in fog-cloud environments for IoT applications depends on task characteristics and fog node characteristics. The Firefly algorithm is inefficient in local search and is not suitable for selecting optimal fog for offloading. The optimal solution is affected by the firefly control parameters, which have large parameters to be tuned.

IV. SYSTEM MODELING AND DESIGN

4.1 System Design and Architecture

The proposed model for air pollution monitoring consists of five entities: IoT devices, fog nodes, trusted authority, proxy server, and cloud server. The system consists of four layers: IoT Devices Layer, Fog Layer, Cloud Layer, and Service Layer. In the IoT devices layer, sensors and actuators are deployed and sensed data. Clusters are formed based on node residual energy level, node degree, and distance between nodes for fast data transmission. The Fog layer allows for verification of data duplication or not, supported by hash generation. In the Cloud Layer, a proxy server maintains a scheduling list for packets transmitted from IoT devices, encrypting and storing them based on sensitiveness. In the Service Layer, sensors information is provided to authorized IoT users.

4.2 IoT Devices Layer

The study uses IoT devices to measure air pollutants concentration in the environment and forward data to a cloud server for processing. It considers air polluting and healthcare-related sensors like CO, NOx, SO2, PM, CO2, and VOC. The devices are authenticated using the Edwards Curve-based Elliptic Curve Algorithm (EC-ECA).

a) Authentication

Trust Authority is a crucial component in intrusion prevention systems, ensuring secure access to cloud user data by generating One Time Signatures for each legitimate user. Cloud users store their data using the Elliptical Curve Cryptography (ECC) method, a public key encryption technique based on elliptic curve theory. ECC generates keys through the properties of the elliptic curve equation, generating keys faster than traditional methods. Each user has a pair of private and public keys, with the public key used for encryption and signature verification and the private key for decryption and signature generation. The ECC algorithm uses shorter keys for higher security levels. The general equation (2) of Edwards Curve E is given as

$$y^2 = x^3 + ax + b \quad (2)$$

Where a, b – Real Numbers, x, y- Points on Elliptical Curve E.

Edwards equation is a non-singular equation that requires a non-singular curve, with characteristic coefficients 'a' and 'b' determining points on the curve. It is explained as shown in equation (3),

$$\Delta = 4a^3 + 27b^2 \quad (3)$$

Where, the value $\neq 0$

Usually, Points on the curve are presented with x and y components similar to Euclidian coordinate system. Equation (4) represents and considers one exception that is one point in the infinity curve representation.

$$A = \begin{pmatrix} a_x \\ b_y \end{pmatrix} \quad (4)$$

The Elliptic Curve Cryptographic Algorithm (ECC) uses a pseudo code to generate a public key using a standard generator P and a random number S. This key is then used for encryption and signature verification, with the primary goal of protecting user data from intruders. The proposed system model for task allocation and secure deduplication

via FaCIIoT is presented, with the system architecture depicted in Fig. 3.

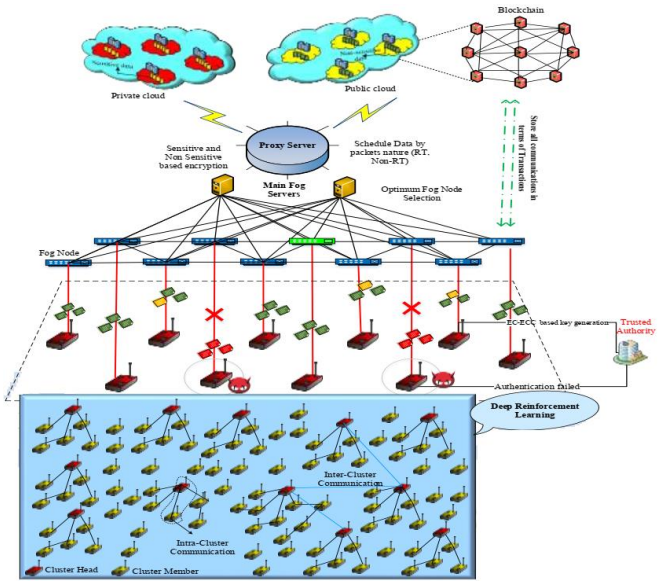

Fig. 3. System Model.

An elliptic curve is a non-singular projective algebraic curve which is presented over some field k with genus 1 and a specified point O. k does not have characteristic 2 or 3, this will be a smooth plane cubic curve with the point at infinity, and the curve as points satisfying the equation (5) and (6).

$$y^2 = x^3 + ax + b \quad (5)$$

Where a & b are discriminant

$$\Delta = -16(4a^3 + 27b^2) \quad (6)$$

The group law on an elliptic curve is exploited for key selection in elliptic curve cryptography is depending upon the elliptic curve as an abelian group with points as elements. The group law is pointing additions which add two points P and Q.

b) Cluster Head Selection and Formation

Sensors are initially grouped based on three parameters: node residual energy, node degree, and distance between nodes. Each cluster has one CH and multiple members, allowing communication within and between CHs for data transmission.

i. Residual Energy (RE)

It represents the current energy level of sensor nodes. As assumed, all nodes have same initial energy (IE) and the energy level is varied over time period. For node N_i the RE is computed as shown in equation (7).

$$RE(N_i) = IE - DE \quad (7)$$

Where $\boldsymbol{DE}$ represents the dissipated energy value over time period.

ii. Node Degree(D)

It defines the connectivity of sensor nodes in the constructed graph. It is computed in terms of number of relative neighbors a node has in the graph

iii. Distance with sink node ($\mathbf{dis(N_i, Sink)}$)

It represents the distance between $\boldsymbol{N_i}$ and sink node. It is computed in terms of Euclidian distance as shown in equation (8).

$$\boldsymbol{dis(N_i, Sink)} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (8)$$

Where (x_2, y_1) and (x_2, y_2) represent the coordinates of N_i and sink node respectively.

iv. Hop count (HP) and mobility (M)

HP defines the number of hops between N_i and sink node. Mobility of the node defines the current mobility speed of the node.

v. Link Stability (LS)

It defines the stability link between N_i and sink node. It can be expressed as shown in equation (9)

$$LS = \frac{Radius}{dis(N_i, Sink)} \quad (9)$$

Where the radius represents the communication range of N_i

vi. RSSI

It is computed in terms of power presented in the radio signal received by N_i from sink node. It computed as shown in equation (10),

$$RSSI = P_0 \left(\frac{dis(N_i, \text{Sink})}{dis_0} \right)^{\sigma} \quad (10)$$

Where P_0 represents the reference power received at the distance d_0 and σ denotes the path loss component. By using all seven metrics leader nodes are selected. At first, the weight value is computed for all nodes based on the RE, D, $dis(N_i, \text{j})$ as shown in equation (11),

$$\text{W}(\text{N}_\text{i}) = \text{RE} + \left(\frac{\text{D}}{\text{dis}(\text{N}_{\text{i,j}})} \right) \quad (11)$$

The nodes are sorted in descending order based on weight value. Then, the threshold value (μ) is computed based on the average weight value as shown in equation (12),

$$\mu = \frac{(W(N_1) + W(N_2) + \cdots + W(N_n))}{n} \quad (12)$$

The nodes which have weight value higher than the threshold ($W > \mu$) are considered for second stage. Thus, the number of nodes to be processed in the next stage is reduced based on the weight value.

In CH, Deep Reinforcement Learning is introduced completely avoid energy consumption issue. Deep reinforcement learning is a new algorithm, which learns and interacts with real-world environment. It is based on the finite Markov decision process (f-MDP). A set of entities used in this algorithm is as follows, S is the set of states, A is the set of actions, the state transition probability $p(S'|s, a)$. It is a probability distribution function on state space for a given action a for state s, the discount factor is $\propto$, which range from 0 and 1, Reward $\gamma = (S * a)$ is computed using state and action (Set of Real Numbers), To get in easier, it is assumed that the rewards are discrete, Use f-MDP when S and A are finite variables.

Assume that the current state s and action a in environment is given, then the probability distribution function for next state s' is computed and also the next reward R is expressed as shown in equation (13),

follows,

$$p(S'|s, a) = P_r(S_{\tau+1} = S'.\gamma_{t+1} = r|S_T = s.A_T = a) \quad (13)$$

The state transition probability is computed according to the reward function (if γ is $discrete$) and it is expressed in equation (14),

$$p(s', r|s, a) = \sum_{r \in Y} p(s', r|s, a) \quad (14)$$

An expected reward is computed for the current s and a as indicated in equation (15),

$$r(s,a) = E[\gamma_{T+1}|S_T = s, A_T = a] = \sum_{r\in Y} r \sum_{s'\in S} p(s', r|s, a) \quad (15)$$

Then the states value functions were defined, which is described by specific policies since future r is based on the agent current actions. In following, the state value function and action value function are computed.

State value function:

The policy π for the state s of value is computed by the expected return, which is represented as $V_\pi(S)$, which is computed towards the current state s. It is computed in equation (16) and (17),

$$V_\pi(S) = E_\pi[G_T|S_T = s] \quad (16)$$

$$V_\pi(S) = E_\pi\left[\sum_{k=0}^{t-T-1} r^k \gamma_{T+K+1}|S_T = s\right] \quad (17)$$

Action Value function:

The policy π for the action Z of value is computed by the action a in state s, which is represented as $Z_\pi(s,a)$. It is computed in equation (18) and (19),

$$Z_\pi(s,a) = E_\pi[G_T|S_T = s, A_T = a] \quad (18)$$

$$Z_\pi(s,a) = E_\pi\left[\sum_{k=0}^{t-T-1} r^k \gamma_{T+K+1}|S_T = s, A_T = a\right] \quad (19)$$

The IDP agent uses a deep reinforcement learning algorithm to update each switch stage based on two metrics: Flow Duration and Packet Inter Arrival Time. Flow duration is the time difference between the first and last packets, while inter packet arrival time is the time difference between two succeeding data packets. The hidden layer computes the weight value and present state of all switches based on input variables. The IDP agent aims to increase the reward obtained from the environment, with the reward function being the major objective function. The IDP agent consists of two goals: assigning benign packet-in messages to switches and avoiding malignant packet-in messages to minimize attack traffic percentage.

c. Secure Deduplication

After authorization of nodes to TA, cluster formation and cluster head selection. The node separates the received data based on the Region ID and then similarity is estimate. Jaccard similarity for computing the similarity between data was used. The formation of the Jaccard similarity is given in equation (20),

$$Sim(P_1, P_2) = \frac{|P_1 \cap P_2|}{|P_1 \cup P_2|} \quad (20)$$

The Jaccard method determines similarity between data packets P1 and P2. If the similarity value is less than $Sim(P_1, P_2)$, the data packets are dropped, and the sensor node ID is included in the packet to notify the sensor that has sensed and provided similar data. If the similarity value is higher, both packets are transmitted to the Cloud Computing Hub (CH) by intermediate sensor nodes. The computation of similarity is processed by intermediate nodes, reducing time consumption. Once redundant data is eliminated, the received data is transmitted to CH, ensuring no redundant data packets are present. SHA-3 is used for duplication verification, ensuring data integrity while transmitting data packets to the cloud server through fog nodes. SHA-3 takes arbitrary input data packets and outputs messages digest or hash values. For a hash generation, 512 bits are used and the properties of SHA-3 are depicted in Table 2.

TABLE 2. PROPERTIES OF SHA-3

SHA-3(512 bits)	
Parameters	Variants
Block size (bits)	576
Capacity	1024
Word size (bits)	64
Rounds	24
Operations	AND, OR, NOR, and NOT
Security strength	256
Output size (bits)	512

After the generation of hashes, CH sends DSC request to near and optimum fog nodes. Fog node checks whether this data packet is received or not. If packets at stored in temporary storage of fog nodes, it will immediately send DSC response. Then CH will store the file to a cloud server via fog nodes.

4.3 Fog Layer

In this layer, fog server selects optimum (nearest) fog node via fog server using EPC algorithm, which follows the procedure of Fast Optimization algorithm.

a) Optimum Fog Node Selection

The EPC optimization algorithm was developed and it is a first-in-its-kind approach to IoT framework improvement. It combines swarm and nature-inspired behavior, organizing penguin behavior through thermal radiation and spiral movement. The algorithm detects an optimal solution in the fourth iteration, reducing mining system execution time. The algorithm takes input from fog nodes and their current status, creating an initial inhabitant's array of the emperor penguin. It computes the fitness function for each node based on current residual energy, distance, and buffer state. Each penguin in the EPC algorithm estimates its own heat radiation, spiral movement, and attractiveness. This approach significantly reduces the execution time of the mining system. In addition, it also determines the new position for the next moving direction.

Table 3 shows the pseudocode process for optimizer operation.

TABLE 3. OPTIMUM FOG NODE SELECTION.

Pseudocode for Optimizer Operation

Require : Optimum fog node
Ensure: < buffer state, energy & distance >:: Optimizer function
Generate population P = $[P_1 \dots P_n]$;
For All P ∈ $[P_1 \dots P_n]$ **do**
Compute → fitness f(i)for P_i;
If (f(i) > f(j))
δ (P_i) ← f(i);
End If;
End For
Emit ($K_n \rightarrow P, V_n \rightarrow \delta(P)$)
End

The heat radiation (P_{hr}) for each penguin is computed using equation (21)

$$P_{hr} = s_a \mathcal{E} \sigma T_a^4 \quad (21)$$

Here, the s_a represents the surface area, T_a denotes the absolute temperature, σ denotes the Stephan Boltzmann constant and $\mathcal{E}$represents the emissivity. The attractiveness (P_A) of each penguin is estimated with aid of the upcoming equation (22),

$$P_{hr} = s_a \mathcal{E} \sigma T_a^4 \; e^{-\mu d} \quad (22)$$

Where, μ denotes the attenuation co-efficient and d represents the distance between the two linear resources. The penguin spiral movement is computed as follows in equation (23) and (24),

$$x_h = a \cos \theta_k \; e^{b\theta_h} \quad (23)$$

$$y_h = a \sin \theta_k \; e^{b\theta_h} \quad (24)$$

Here, x_h and y_h indicates x and y components of the penguin position 'h'. The spiral moving behaviour of the penguins in the EPC algorithm provides the searching speed effectually. With the use of aforesaid expressions, EPC estimates the fitness function for each penguin which is signified as follows in equation (25),

$$f(i) = \sum_{i=1}^{n} P_{hr} P_A x_h y_h \quad (25)$$

Using the above equation, fitness function is estimated for each penguin which defines the optimal value with less amount of time. Since, it converges fastly with optimal solution compared to the other traditional algorithm like PSO, GA and so on.

b) Packet Scrutinization in Fog

The fog server is crucial in the intrusion detection phase, collecting packets from IoT devices at different locations. Fog nodes monitor packet behavior and assign threshold values to each node. Due to dynamic device movement, packet traffic arises in the fog environment. When packet arrival exceeds the threshold, fog nodes migrate from heavy traffic to idle nodes. The fog node with packet algorithm is shown in Fig. 4.

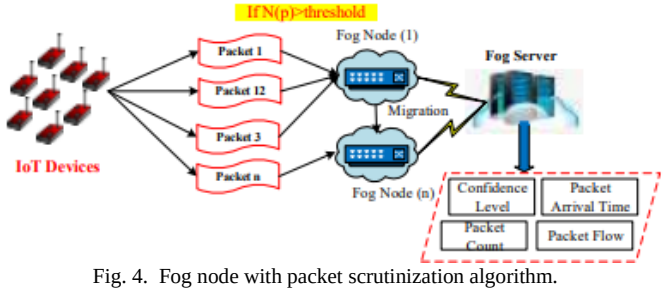

Fig. 4. Fog node with packet scrutinization algorithm.

The proposed packet scrutinization algorithm analyzes collected packets by fog servers based on packet arrival time, packet flow, packet count, and confidence level. The algorithm considers packet arrival time, packet flow, trust value, and packet counting based on headers. The confidence level is the frequency of attribute appearances in packet flows, while packet count is determined by the number of packets in the sequence. The confidence level is calculated based on single attribute and pair of attributes follows as in equation (26),

(i) Confidence level for single attributes,

$$C(A_i = a_{i,j}) = \frac{N(A_i = a_{i,j})}{N_n} \quad (26)$$

where, i = 1,2,3…n and j = 1,2, 3... m_i

(ii) Confidence level for attribute pairs,

$$C(A_{i1} = a_{i1,j1}, A_{i2} = a_{i2,j2}) = \frac{N(A_{i1} = a_{i1,j2}, A_{i2} = a_{i2,j2})}{N_n} \quad (27)$$

where, i_1 = 1,2,3…n, i_2 = 1,2,3…n, and j_1 = 1,2, 3. .m_1, j_2 = 1,2,3...m_2, N – the number of attributed that are considered to overcome the folding attacks and DDoS attacks, N_n-Total number of packets on packet flow in one time interval. $A_i - i^{th}$ attribute in packet, $N(A_{i1} = a_{i1,j2}, A_{i2} = a_{i2,j2})$ Number of packets whose attribute Ai1 has the value ai1, j1 and Ai2 422 has the values ai2, j2 423 in packet flow in one time interval (t). Using above equation, calculate the confidence level for every packet. If confidence level of the packet is low, then the corresponding packet is discarded. Fog node only allows the packets which have high confidence level than threshold level. Fig. 5 shows the clear view about the working process of packet scrutinization algorithm.

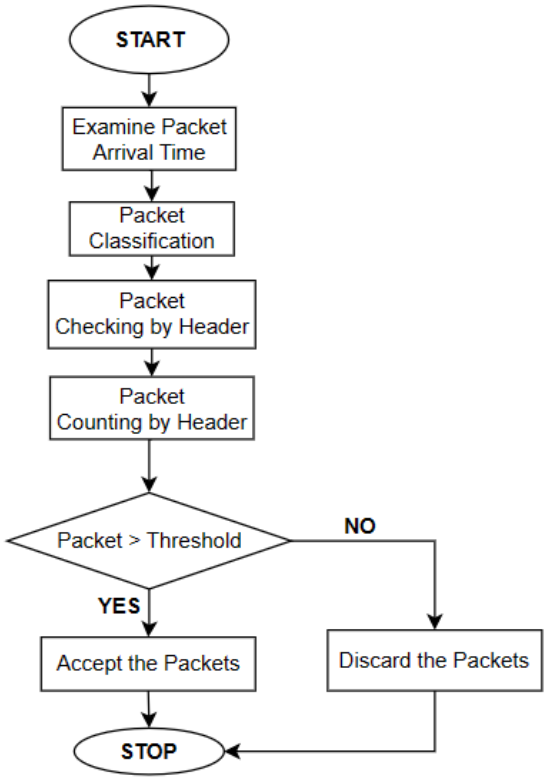

Fig. 5. Flowchart for packet scrutinization algorithm.

Table 4 shows the clear view about the working process of packet scrutinization algorithm.

TABLE 4. PROCESS OF PACKET SCRUTINIZATION ALGORITHM.

Algorithm for Packet Scrutinization
Step 1: Start Step 2: Examine the arrival time of every packet from all cloudlets

Step 3: Classify the packets based on arrival time and its flows Step 4: Check packets according to its header Step 5: Count the packets according to its header Step 6: Check confidence level if (packet > threshold level) Accept packet Else Discard packet Step 7: End

Thus, using this algorithm, we can easily detect and remove the initial flooding attack and port scanning attack.

4.4 Cloud Layer

A proxy server is deployed before cloud servers, constructing queues for data forwarded by IoT devices and acting as the primary node between the fog node and cloud server.

a) Queue Modeling

After classification, the normal packets are further processed into fog node whereas the intruder packets are deleted from the fog node. In order to provide efficient processing to the packets, the research work introduces a queue modeling named M/M/C which performed based on the packet prioritization. Usually queuing system is characterized with four basic components such as Queue Discipline, Arrival rate, Service Channel and Service Rate The proposed queuing model estimates the arrival time based on packet entering into environment. Then service channels are specified with multiple s that can estimate different packets and service rates are defined as that multiple packets are executed with different Proxy Servers at a time. The prioritization of the packet is based on the type of rules which are represented with processing time and arrival time. Fig. 6 illustrates the proxy server allocation process performing by the $M/M/C$ queue modelling.

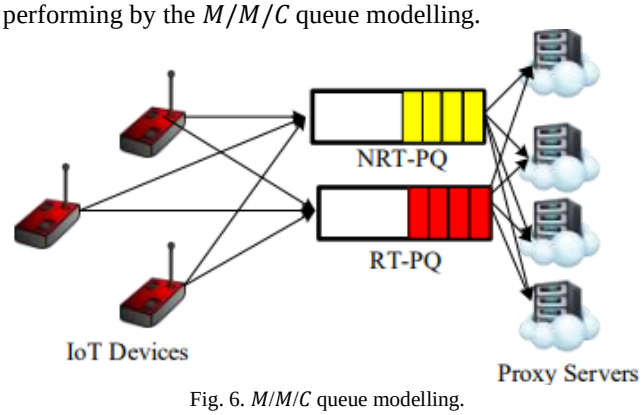

Fig. 6. $M/M/C$ queue modelling.

Fig. 7 shows the working process of $M/M/C$ queue modelling. In this method, multi-users and multi-servers are involved to allocate the packets in specific virtual machine for further execution.

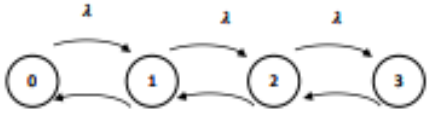

Fig. 7. State space diagram of $M/M/C$ Queue modelling.

Where λ refers to packet arrival time and 2μ represents the service rate of the packets. Our proposed queue model has multi-user packets (∞) and multi-server (C) "$M/M/C$" and we propose four priorities such as A, B, C and D which is depicted as follows:

(i) **Class 1:** When a packet has short waiting time and its request type is urgency then packet gets the first priority on queue for processing.

(ii) **Class 2:** A packet with long waiting time and has urgency request then we furnish the second priority for the packet.

(iii) **Class 3:** A packet with short waiting time and has no urgency gets the third priority for processing.

(iv) **Class 4:** A packet with long waiting time and has no urgency for processing, then we furnish the fourth priority for the packet.

Based on these conditions, the research work allocates the normal packets to the virtual machine to further execution that improves the QoS of our proposed system.

The processing steps of queue modelling (Table 5), involves allocating normal packets to the virtual machine for further execution, thereby improving the Quality of Service (QoS) of the proposed system.

TABLE 5. PROCESSING STEPS OF QUEUE MODELLING

Steps for Queue Modeling
Step 1: Start Step 2: Proxy Servers Step 3: if (P is RT: (Urgency && WT-short) PT: "Class 1" End if Step 4: if (P is RT: Urgency && WT-long) PT: "Class 2" End if Step 5: if (P is RT: No Urgency && WT-short) PT: "Class 3" End if Step 6: if (P is RT: No Urgency && WT-Long) PT: "Class 4" Step 7: $P \rightarrow Q$ Step 8: $Q \rightarrow$Proxy Servers Step 9: End

Where, P is the packet, Q represents FIFO queues, RT specifies request type; WT denotes waiting time and PT illustrates priority type (1, 2, 3, and 4) of packets.

b) Sensitive & Non-Sensitive Data Encryption

Data encryption is the first security process, dividing message blocks into two sub-blocks, one encrypted with AES and the other using RC-6 algorithm. This process converts plain text into cipher text using keys. The proposed encryption splits sensor node texts into two blocks, each 128 bits long, for secure storage.

First Block

Let p_i [0: N/2-1] and P_i [N/2: N-1] be the two divided block for the plain text, here N is not an integer number which has a fraction. From this division the first block N/2 is encrypted using AES as mentioned above. The size of this block is 128 bits and having the generated key K and length L as in equation (28) and (29).

$$p_i = \sum_{i=0}^{i=\frac{n}{2}-1} B_i \; 0 \le i \le \frac{n}{2} - 1 \qquad (28)$$

$$C_i = e_{AES}(K, B_i) \qquad (29)$$

Where the plain text of the first block is converted into cipher text that is denoted as C_i and e_{AES} is the encryption function that is the function used in AES algorithm. This encryption is followed with the processing of next 128-bit block. Here the entire data packets are split into equal halves for easier and faster processing of data. This division enabled to provide security of the data.

Second Block

For the second block of the plain text, followed by RC-6 encryption to secure the data. The second block P_i [N/2: N1] using RC-6 encrypts the data at faster speed with the growth of the security level as shown in equation (30) and (31).

$$p_i = \sum_{i=n/2}^{i=n=1} B_i \quad \frac{n}{2} \leq i \leq n-1 \tag{30}$$

$$C_i = e_{RC6}(K, B_i) \tag{31}$$

The second 128-bit block is encrypted using two cipher texts for single plain text, providing security. The Fiestel algorithm is used for non-sensitive data encryption, splitting the data into two parts to improve security. This symmetric technique is used in cryptography to construct block ciphers, making encryption and decryption operations similar. The proposed Fiestel algorithm splits the data into two parts, enhancing the security of the encrypted data. The Fiestel encryption scheme encrypts data at a fast rate, with multiple rounds of handling raw data, each with a substitution process monitored by the permutation process. The process is virtually identical in structure, ensuring efficient and secure data storage.

Let $\mathbb{F}$ be the round function of the Fiestel cipher and K_0, K_1, ... K_n be the sub-keys for the rounds 0, 1...n respectively. At first, data S_{ib} is split into two equal pieces that are S_{ibL} and S_{ibR}. For each round r=0, 1…. n compute, as indicated in equation (32) and (33)

$$S_{ibL}(r+1) = S_{ibR}(r) \tag{32}$$

$$S_{ibR}(r+1) = S_{ibL}(r) \oplus \mathbb{F}(S_{ibR}, K_i) \tag{33}$$

Where $\oplus$ represents the XOR operator and K_i represents the key value. Then the cipher text attained as S_{ibR+1} and S_{ibL+1}. The one of the advantageous of Fiestel encryption scheme is round function is doesn't need to be invertible.

4.5 Services Layer

In this layer, IoT data are retrieval after the verification of user's authentication. If registration is successful for users in TA, then the searching result is provided for users.

The procedure of Blockchain

Step 1: IoT user/ device requests for a transaction from Blockchain.

Step 2: A new block that denotes the particular transaction is created.

Step 3: Then the particular block will be disseminated to all the other nodes participating in the network.

Step 4: Further all the nodes that received the transactions, will validate the currently received transaction.

Step 5: After verification, the particular block is included into the Blockchain

Step 6: later the transaction is verified and executed.

Using asymmetric cryptography of elliptic curve, pair of keys is generated as public key and private key for IoT device. This PoW is a blockchain authentication method followed in blockchain. This consensus algorithm is more helpful in supporting resource constrained devices. The use of asymmetric cryptography in blockchain enables to provide incorruptible data storage in the blockchain network. The use of PoW is approximately 200 times faster. The blocks in the blockchain are authenticated using PoW that is represented in Fig 8. The nodes present in the network are enabled to record the distributed ledger and so the transaction information of the nodes can be followed properly.

Each block is authenticated and for every successful validation the node will be credited increment in its trust values. Here, the use of asymmetric ECC for key generation with 256 bits attains 128 bits in security level which is effective and it sustains to be protected in the system.

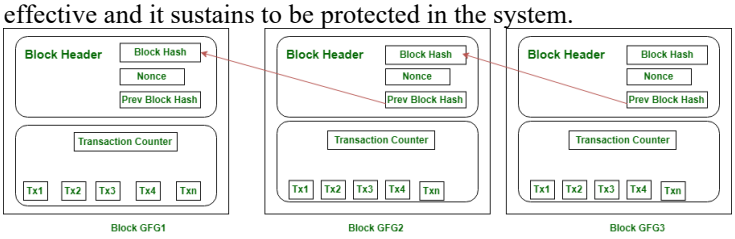

Fig. 8. PoW Consensus.

V. RESULTS AND DISCUSSION

In this section we well described the simulation part and also discussed the performances of the proposed scheme using several metrics. Table 6 shows the system configuration parameters, Table 7 indicates network environment parametric values, and Table 8 shows packet information parameters.

TABLE 6. SYSTEM CONFIGURATION

Name	Description
Simulation Tool	NS-3.26
Development toolkit	JDK-1.8
Operating System	Ubuntu 14.04LTS
Development Platform	NetBeans 8.0
Processor	Pentium (R) Dual-core CPU E5700@3.00 GHs
Installed memory	2GB RAM

TABLE 7. NETWORK ENVIRONMENT

Simulation Parameters	Values
Number of nodes	100 IoT devices
Number of fog nodes	5
Number of cloud server	1 (Hybrid cloud)
Number of simulation tasks	10, 20, 30, 40, and 50
Number of smart gateways	1
Simulation area	1000m x 1000m
Task arrival rate	[0,5]
Simulation time	100 seconds
Initial energy of a node	5J
Traffic type	CBR
Packet interval	0.1s
Learning rate	0.2

TABLE 8. PACKET INFORMATION

Mobility Configuration Metrics		
Mobility of Mus **Mobility model of MU** **Interval Time**	300ms Random way point model 0.1s	
Packet Configuration metrics		
Packet interval **Bit rate**	100ms 2Mbps	
Attack Configuration metrics (PPP-Packets per second)		
Attack rate (per Attacker)	High(pps)	1000
	Low (pps)	20
Cumulative Attack Packet Rate	High(kbps)	1000-1200
	Low(pps)	6-70
Cumulative Traffic Rate	High (mbps and pps)	3.6
	Low (mbps and pps)	6-70kbps
Protocol Configuration Metrics		
Protocol Used **Latency (processing)**	IPv6 10μs	
Deep Reinforcement Learning Configuration Metrics		
Reward Rate	0.9	
Batch Size	100	
Learning rate	0.001-0.1	
The number of hiden layer	3	
The number of nodes at input layer	2	
The number of nodes at output layer	2	
Activation function	ReLU and Linear	
Optimizer	ADAM	

5.1 Simulation Environment

This section presents a simulation of a proposed model using NS3 with Java, an open-source Java-based network simulator developed by the CLOUDS laboratory at the University of Melbourne. NS3 is useful for resource management in IoT, edge computing, and fog computing paradigms. The simulation involves a large number of IoT devices and fog nodes. The proposed Cloud-IoT environment includes IoT devices, data users, gateway devices, cloud servers, and TA. IoT devices sensing their surroundings and encrypting data using lightweight encryption. Encrypted data is stored in a cloud server via gateway devices, with TA providing security by allowing only authorized users to access the data. Fig. 9 shows simulation details while Fig. 10 shows, blockchain details in simulation environment.

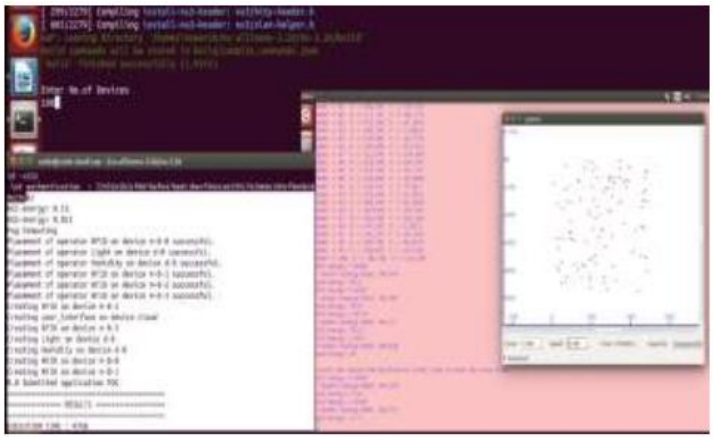

Fig. 9. Simulation details.

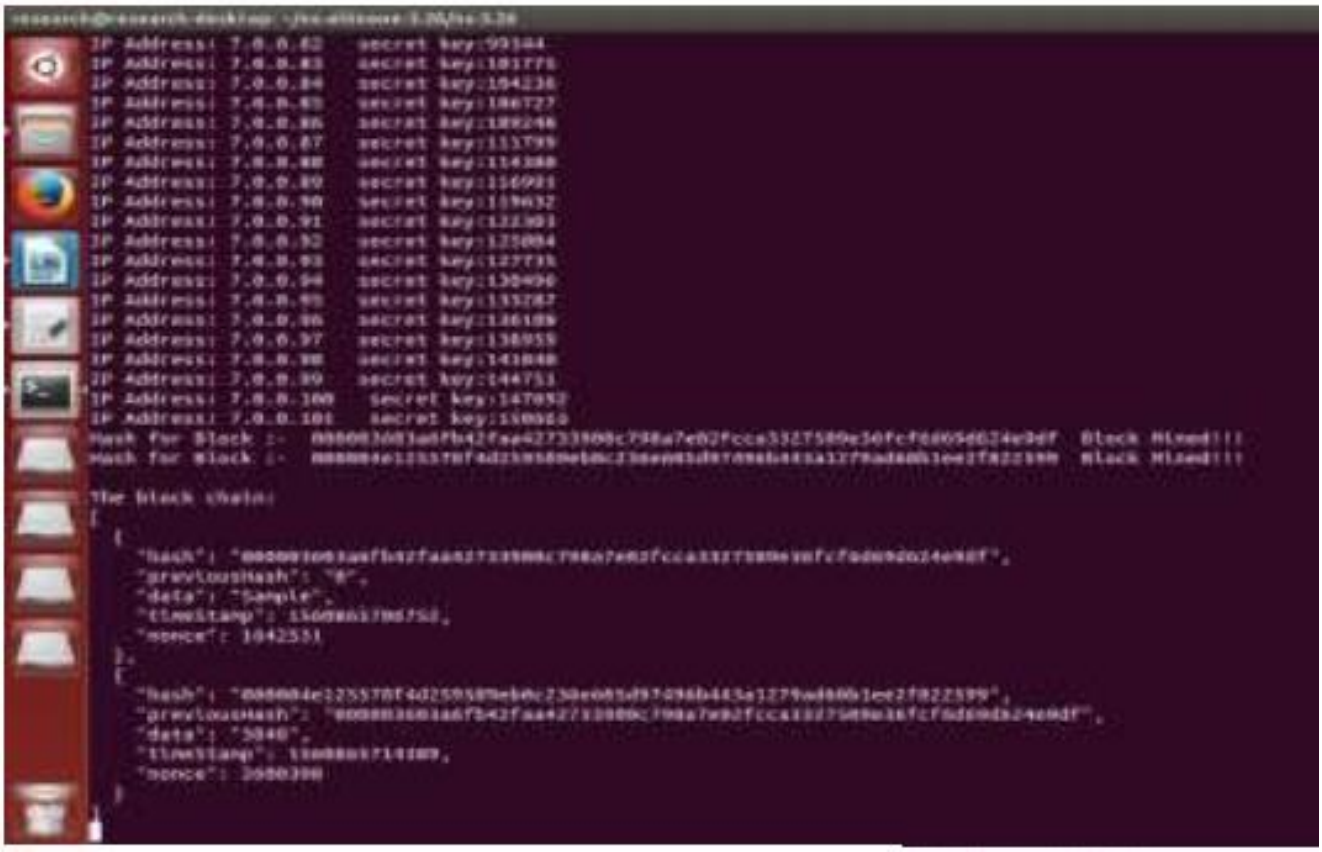

Fig. 10. Blockchain details

5.2 Case Study: Remote Health Monitoring System

The proposed scheme for Remote Health Monitoring is tested to address the issue of duplicated data transmission in fog environments, particularly in industries and vehicles near cities. This duplication leads to poor quality of service and severe cloud service issues. Healthcare information must be kept private for storage and retrieval, reducing environmental harm. The fog-enabled IoT system uses sensors to monitor environment-based and healthcare-related information, with a simulation topology for healthcare monitoring in IoT shown in Fig. 11.

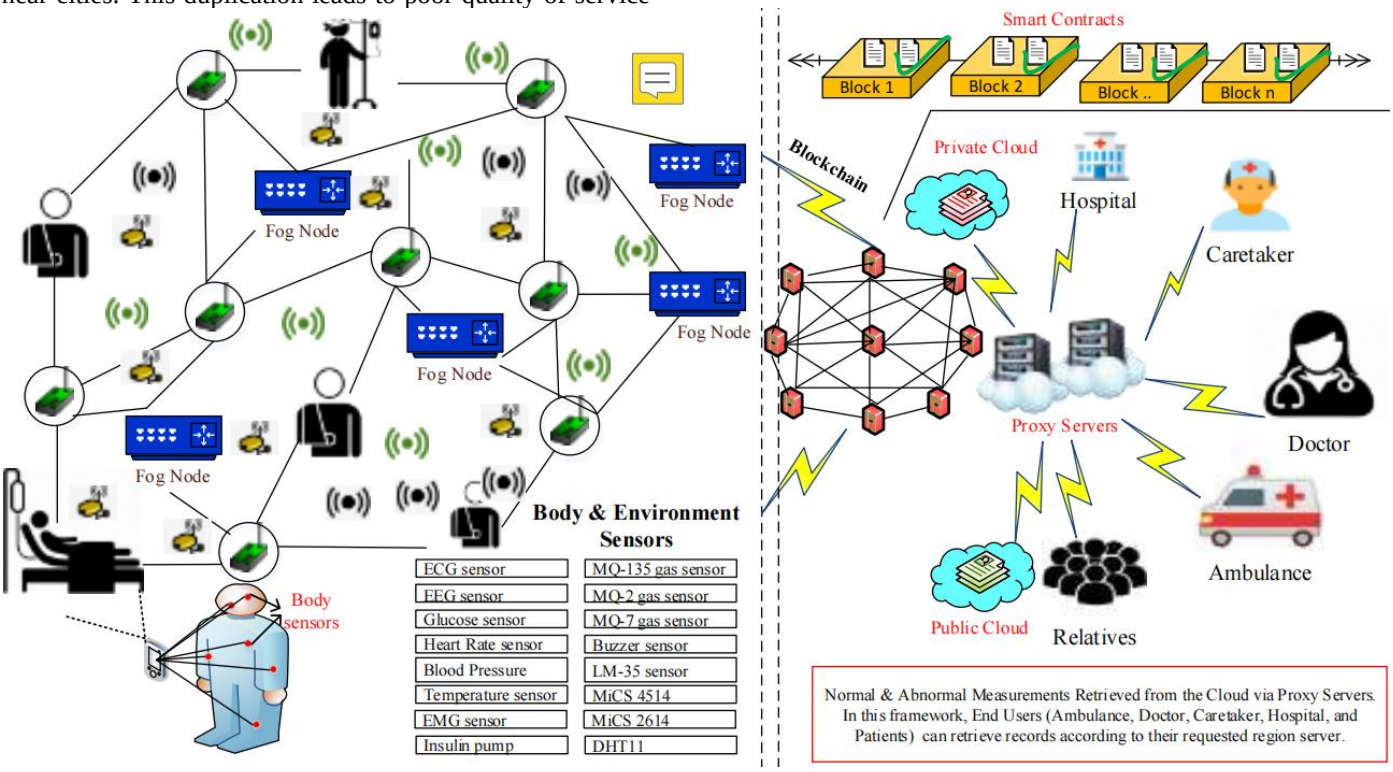

Fig. 11. Blockchain Simulation Topology in health monitoring.

In this work we categorized air pollutants into three classes that are given below:

i. **Primary air pollutants**

Generally, it is produced from several gas sensors include carbon dioxide, sulfur oxides, nitrogen oxides, carbon monoxide, volatile organic compounds, radioactive pollutants, etc

ii. Secondary air pollutants

It is generated by communications made from primary air pollutants include ground level ozone, peroxyacetyl nitrate, smog, etc.

iii. **Others**

It covers minor hazardous and organic persistent air pollutants. Table 9 and Table 10 show list of IoT sensors used in this paper for monitoring remote healthcare monitoring system and their functionalities. Sensors are MQ-135, MQ-2, MQ-3, Buzzer sensor, LM-35, MiCS4514, MiCS2614, and DHT11 are deployed for measuring environment information in this area. Here, the body sensor nodes are deployed in IoT layer and the aggregated sensed data is transmitted to data processing unit. The body sensors can be heartbeat sensor, temperature sensor, pressure sensor, oxygen level sensor and motion sensor. Then the processed data is delivered to doctors, caretakers, ambulance and relatives based on severity level. This intelligent healthcare system mitigates all related issues in conventional healthcare such as delay, inaccurate system and so on.

TABLE 9. SENSORS AND FUNCTIONALITIES

Environment Sensor Type	Functionality
MQ-135 gas sensor	Measuring air quality
MQ-2 gas sensor	To detect CO, Alcohol, Smoke/Propane, H2, LPG, and CH4
MQ-7 gas sensor	Detecting CO, and suited sensing concentrations CO in the air
MQ-3 gas sensor	Detect Benzine, Hexane, CO, Alcohol
Buzzer	Giving alarm to inform about unhealth Air or Exceeding chemical values on each sensor
LM-35 sensor	For measuring temperature inputs
MiCS 4514	For measuring NO2 and CO
MiCS 2614	For measuring O3
DHT11	For measuring Humidity and temperature

TABLE 10. SENSORS AND FUNCTIONALITIES

Body Temperature type	functionality
ECG sensor	**For measuring heart rate**
EEG sensor	To detect brain actions and recording nerves activities
Glucose sensor	Detecting the glucose content in the body
Heart rate sensor	Detects heart rate accurately
Body pressure sensor	Detects blood pressure level
Temperature sensor	Measuring temperature inputs
EMG sensor	Measuring muscles information
Insulin Pump	Measuring pumps

5.3 Evaluation Measures

The integration of IoT, fog, and cloud computing paradigms significantly impacts average latency, user satisfaction, network lifetime, energy consumption, and security strength.

i. Average Latency

It is the time required to respond to the user's given request at a time. The average latency is defined as the sum of time taken to process all requests given by the IoT device as shown in equation (34),

$$A_l = \min + \frac{\max}{2} \quad (34)$$

Where Al is the average latency and its unit is milliseconds (ms). It is computed on minimum and maximum amount of time. Minimum time is zero and the maximum time is the time require for processing single request.

ii. User Satisfaction

It is a metric that finds how well a service response from the fog/cloud will satisfy user's requirement. It is not same for users with requests specific service. Hence it differs based on user's service request arrival time and distance to the fog/cloud servers as shown in equation (35),

$$U_S = S_{RT} + S_{IT} + S_Q \quad (35)$$

Where Us is the user satisfaction, SRT is the service response time, SIT is the service-initiated time, and SQ is the service quality.

iii. Network Lifetime

It is defined as the amount of time during which the sensor network is fully operative. It can be defined as the maximum duration of operational time of the network while the network performs specific task. It is expressed in (36),

$$NL = \frac{E_O - E[UU]}{P + \delta E[Rep]} \quad (36)$$

Where $E0$ represents the initial energy consumed by all sensor nodes, $E[UU]$ is expected wasted energy, $E[Rep]$ represents the expected reporting energy and δ is the average sensor reporting rate. The network lifetime is measured in time duration or in number of rounds.

iv. Energy Consumption

It is defined as the amount of energy consumed to perform processes such as sensing, data transmitting and data receiving. Energy consumption of the network is represented as follows in (37)

$$E_C(N) = \sum_{i=1}^{n} \left[E_{Tx}(N_i) + E_{Rx}(N_i) + E_{sensing}(N_i)\right] \quad (37)$$

Where ETx, ERx, $Esensing$ energy consumed for transmission, reception and sensing by a sensor node N_i.

v. Security Strength

It is essential to analyze the designed system for task allocation and deduplication. It ensures the user satisfaction and supported for massive data storage at cloud servers. It is computed by the following (38).

$$S_{st} = K_s + M_s + E_t + D_t \quad (38)$$

Where Sst is the security strength, Ks is the key size, Ms is the message size, Et is the encryption time, and Dt is the decryption time.

vi. Detection rate (D)

The detection rate (D) is defined as the percentage of correctly detected attack records of the intrusion detection system. It also defined as the ratio between numbers of attack detected in the system to the number of attacks appeared in the system as in (39).

$$\mathrm{D} = \frac{\text{Number of detected attacks}}{\text{Number of attacks present}} \text{ X } 100\% \quad (39)$$

Detection rate in terms of TP and FN is shown in (40)

$$\mathrm{D} = \frac{\mathrm{TP}}{(\mathrm{TP} + \mathrm{FN})} \quad (40)$$

vii. Speedup ratio

The speed up ratio is defined as that supports increasing the performance between two nodes. Here the average execution time take for training the data is considered as speed up ratio (SR) as shown in (41).

$$\mathrm{SR} = \frac{\text{Time taken for intrusion detection}}{\text{Overall number of packets}} \text{ x } 100\% \quad (41)$$

viii. Throughput

Throughput (T) is defined as the rate that packet finishes successful processing of packets. It is also defined as the ratio between the number of successfully executed packet to the total number of packets in the intrusion detection system as shown in (42).

$$\mathrm{T} = \frac{\text{Numof successfully processed packets}}{\text{Total number of packets}} \text{ x } 100\% \quad (42)$$

ix. Packet loss ratio

Packet loss ratio (PLR) is defined as the ratio between the numbers of packets lost to the total number of packets sent in the intrusion detection system as shown in (43).

$$\mathrm{PLR} = \frac{\text{Number of lost packets}}{\text{Number of sent packets}} \text{ x } 100\% \quad (43)$$

5.4 Comparative Analysis

As earlier mentioned, different performance evaluation measures are evaluated and compared with four previous

works, namely, task allocation and secure deduplication (TA & SD) [40], fog-based energy efficient routing protocol (FEER) [41], adaptive block compression sensing (ABCS) [39], and secure data storage and searching in IoT (SDSSIIoT) [42].

a) Average Latency

IoT applications like fire accidents and healthcare require high latency constraints of 10s of ms. By enabling data collection and processing features like clustering and classification at the device layer or fog layer, latency can be reduced, as shown in Fig 12.

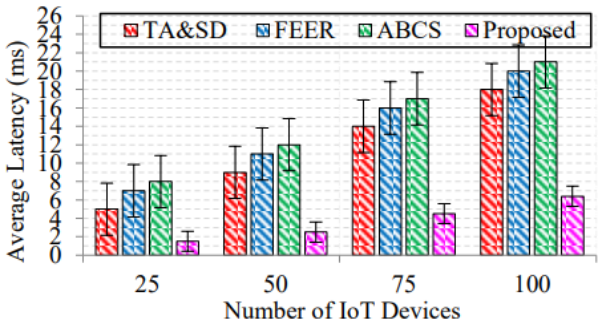

Fig. 12. Average latency vs. No. of devices.

The proposed fog layer in the cloud layer aims to minimize latency for IoT devices, compared to previous mechanisms like TA&SD, FEER, and ABCS. The average latency for twenty IoT devices is 2.7ms, a minimum compared to previous works' 8.9ms, 10.54ms, and 11.54ms. TA&SD requires minimum average latency due to avoiding redundant copies in the fog layer. The proposed scheme uses EPO for optimal fog node selection and SHA-3 (512bits) for hash generation, eliminating duplicate data in the fog layer. This approach reduces waiting time for data transmission and collection.

b) User Satisfaction

The system's performance is attributed to the optimal service provided to users, achieving the best QoS, which is determined by each QoS parameter. Fig. 13 shows the performance of user satisfaction with the number of IoT devices is crucial, as it impacts the performance of QoS parameters.

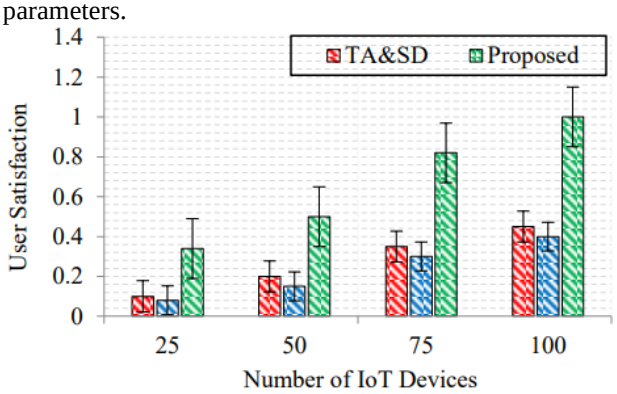

Fig. 13. User satisfaction vs. No. of devices.

The proposed scheme offers low service response time and high service quality, with an average satisfaction rate of 0.5, surpassing TA&SD (0.21) and SDSS-IIoT (0.17). However, the lower satisfaction rate in TA and SD is due to improper cloud server management and increased latency.

c) Network Lifetime

The network lifetime is the maximum operating time of nodes in a network for a specific task. The proposed protocol increases the network lifetime by up to 100 nodes, as the number of sensor nodes increases. The network lifetime metric is inversely proportional to energy consumption, and reducing energy consumption impacts the network lifetime. The proposed scheme reduces energy consumption and improves network lifetime. In FEER, network traffic is reduced, scalability is improved, and latency is minimized, but network lifetime is less due to routing among fog nodes. In contrast, TA&SD has less network lifetime due to poor fog node selection, which leads to high energy consumption at IoT devices. The authors did not focus on optimal fog node selection, which is generally resource-constrained. Overall, the proposed protocol improves network lifetime and reduces energy consumption in IoT devices. Improve network lifetime in fog assisted IIoT is a challenging task. In this paper, the network lifetime for comparison was evaluated. Fig 14 shows the performance of the network lifetime with respect to number of IoT devices.

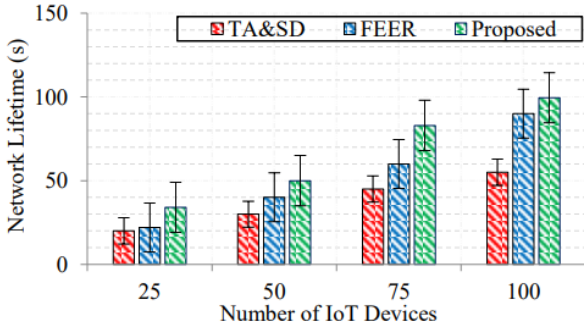

Fig. 14. Network lifetime vs. No. of devices.

c) Energy Consumption

Fog-enabled IoT applications face high energy consumption due to energy constraints in sensors, devices, and actuators. To address this, energy-efficient tasks like clustering can be proposed, which reduces mathematical computations and requires minimal energy consumption in IoT devices. Fig. 15 illustrates the performance of energy consumption based on the number of devices.

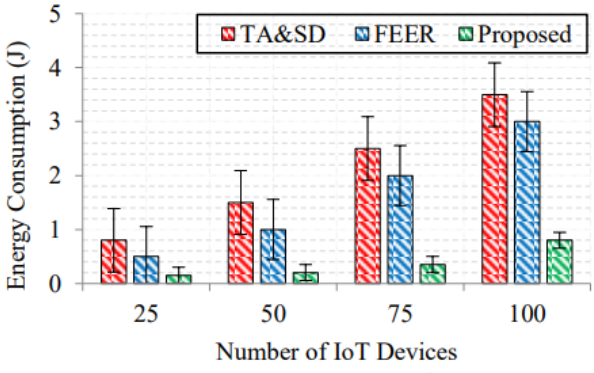

Fig. 15. Energy consumption vs. No. of devices.

The proposed scheme for task allocation and secure deduplication in FaCIIoT has significantly reduced energy consumption compared to previous works like TA&SD and FEER. FEER's ACO-based routing consumes more energy and lacks assurance for the shortest path. TA&SD verifies data deduplication at fog nodes randomly, requiring large

computations. The proposed scheme's average energy consumption is 0.26J, compared to 1.5J and 1.21J for TA&SD and FEER, respectively. The system architecture, which includes best fog node selection and cluster formation, results in less energy consumption.

d) Security Strength

The study examines the security of real-time applications, particularly in healthcare monitoring in IIoT. It compares the performance of a proposed scheme with previous works based on key size, message size, encryption, and decryption time. The results show that the proposed scheme has better security strength as shown in Fig. 16, compared to previous works like TA&SD and SDSS-IIoT, which were found to have less security strength due to ineffective security algorithms as illustrated in Table 11.

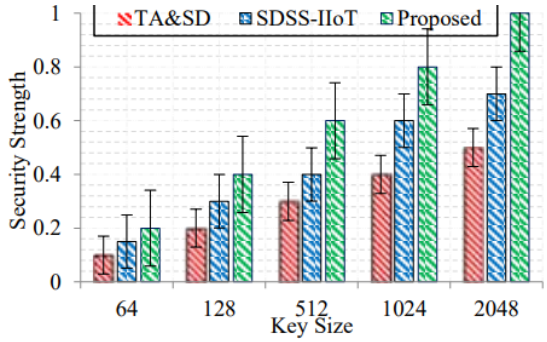

Fig. 16. Security strength vs. key size.

TABLE 11. SECURITY STRENGTH FOR PROPOSED VS. PREVIOUS WORKS

Key size	Security Strength		
	TA & SD	SDSS-IIoT	Proposed
64 bits	0.1	0.15	0.2
128 bits	0.2	0.3	0.4
512 bits	0.3	0.4	0.6
1024 bits	0.4	0.6	0.8
2048 bits	0.5	0.7	1.0
Average	**0.25**	**0.358**	**0.5**

In [41], A secure KNN algorithm in SDSS-IIoT was proposed to improve data confidentiality, increase storage capacity, and prevent privacy data leakage, addressing the weaknesses of the BLS-Pseudorandom function. Fig 17 and Table 12 shows the performance of the security strength with respect to message size (bits). The average security strength for the proposed scheme is 0.61, which is higher than previous works such as TA&SD, SDSS-IIoT since its obtained 0.341, and 0.358, respectively. We proposed ECC-HM, which is lightweight cryptographic algorithm, which gives high security strength when message size increases. It consumes minimal amount of time for encryption and decryption.

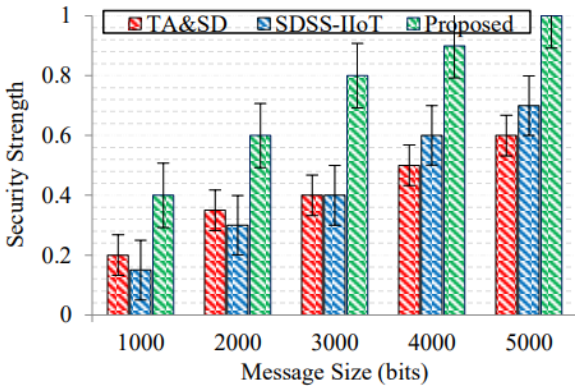

Fig. 17. Security strength vs. message size.

TABLE 12. SECURITY STRENGTH VS. PREVIOUS WORKS

Message size	Security Strength		
	TA & SD	SDSS-IIoT	Proposed
1000 bits	0.2	0.15	0.4
2000 bits	0.35	0.3	0.6
3000 bits	0.4	0.4	0.8
4000 bits	0.5	0.6	0.9
5000 bits	0.6	0.7	1.0
Average	**0.341**	**0.358**	**0.61**

Fig. 18, 19, Table 13, and Table 14 show the performance of security strength with respect to time of encryption and decryption. Pseudorandom function and secure KNN algorithm are not lightweight cryptography and thus it takes high processing time.

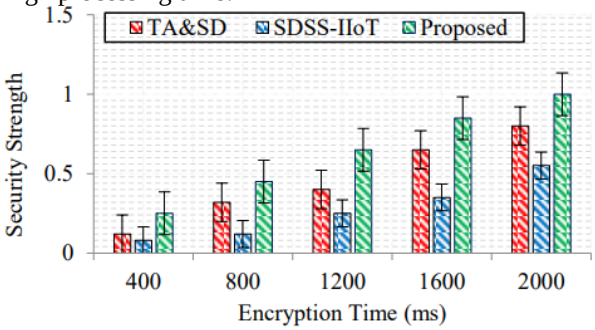

Fig. 18. Security strength vs. Encryption time.

TABLE 13. SECURITY STRENGTH OF THE PROPOSED VS. PREVIOUS WORKS

Encryption Time	Security Strength		
	TA & SD	SDSS-IIoT	Proposed
400ms	0.12	0.08	0.25
800ms	0.32	0.12	0.45
1200ms	0.4	0.25	0.65
1200ms	0.65	0.35	0.85
2000ms	0.8	0.55	1.0
Average	**0.381**	**0.225**	**0.533**

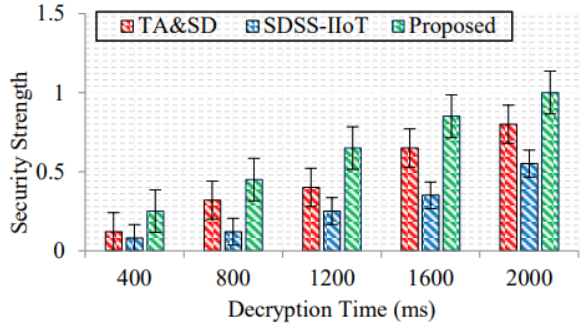

Fig. 19. Security strength vs. decryption time.

TABLE 14. SECURITY STRENGTH OF THE PROPOSED VS. PREVIOUS WORKS

Decryption time	Security Strength		
	TA & SD	SDSS-IIoT	Proposed
400ms	0.06	0.04	0.02
800ms	0.16	0.06	0.03
1200ms	0.2	0.12	0.08
1600ms	0.35	0.15	0.011
2000ms	0.4	0.55	0.012
Average	**0.195**	**0.153**	**0.0255**

e) Detection Rate

This paper presents a new algorithm for classifying packets into normal or attack types, identifying frequent or rare attacks. The model compares with previous works in fog Cloud environments and can be adjusted based on packet and node arrival. The model achieves a high detection rate of 99.4% for any class, surpassing previous works' averages of 97.5%, 94.52%, and 97.86%. The model's performance is compared to previous works in fog Cloud environments as shown in Fig. 20.

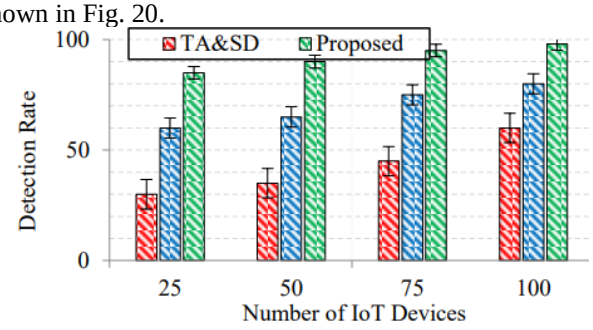

Fig. 20. Detection rate vs. number of IoT devices.

This paper proposes a method for intrusion prevention using trusted authority, a one-way hash function, which restricts malicious node access, improving detection rate (DR) against presence attackers, as legitimate nodes can be easily compromised.

f) Throughput

It is defined as the successful packets transmission rate than previous works. It is a positive indicator so it must be higher to show the system has obtained better performance. Fig 21 shows the result for throughput with respect to number of nodes.

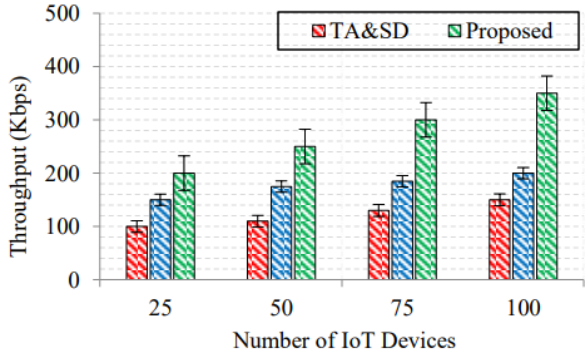

Fig. 21. Throughput vs. Number of IoT devices.

A novel approach to attack detection using a DNN model, combining lightweight algorithms for effective classification was presented. This model outperforms previous works in terms of average latency, user satisfaction, network lifetime, security strength, and energy consumption. The model achieves an average throughput of 220kbps, surpassing previous works such as TA & SD, FEER, ABCS, and SDSS-IIoT. The paper addresses research questions on efficient fog node allocation to different IoT users, proper storage file organization to reduce energy consumption and delay, protection of the entire Fog-assisted Cloud-based IoT environment against attackers, and proposing a robust blockchain-based IoT architecture for task distribution and secure deduplication.

VI. CONCLUSION AND FUTURE WORKS

Healthcare is a significant issue in the industrial sector, with deduplication being crucial to minimize storage capacity and latency of cloud servers and fog nodes. To address this, a paper was designed using a Fog assisted Cloud environment for IIoT, focusing on Task Distribution and Secure Deduplication. The optimal CH was selected for task distribution from IoT devices to fog nodes, using SHA-3 for aggregated data. A proxy server was deployed between cloud servers and fog nodes to schedule user queries. Lightweight algorithms were proposed for data encryption to ensure data confidentiality. Simulations were conducted to compare the proposed scheme with previous works, revealing it outperforms them. The paper plans to focus on real-time applications and use fault detection mechanisms for error-correction in the future.

VII. REFERENCES

[1] Aazam, M., Zeadally, S., & Harras, K. A. (2018). Deploying Fog Computing in Industrial Internet of Things and Industry 4.0. IEEE Transactions on Industrial Informatics, vol. 14, no. 10, pp. 4674-4682, 2018

[2] Sharma, S., & Saini, H. (2019). A Novel Four-Tier Architecture for Delay Aware Scheduling and Load Balancing in Fog Environment. Sustainable Computing: Informatics and Systems, 100355.

[3] Wu, C., Li, W., Wang, L., & Zomaya, A. (2018). Hybrid Evolutionary Scheduling for Energy-efficient Fog-enhanced Internet of Things. IEEE Transactions on Cloud Computing, 1–1.

[4] Haider, F., Zhang, D., St-Hilaire, M., & Makaya, C. (2018). On the Planning and Design Problem of Fog Computing Networks. IEEE Transactions on Cloud Computing, 1–1.

[5] Verma, P., & Sood, S.K. (2018). Fog Assisted-IoT Enabled Patient Health Monitoring in Smart Homes. IEEE Internet of Things Journal, 5, 1789-1796.

[6] Dautov, R., Distefano, S., & Buyya, R. (2019). Hierarchical data fusion for Smart Healthcare. Journal of Big Data, 6, 1-23. [7] Naghshvarianjahromi, M., Kumar, S., & Deen, M.J. (2019). Brain-Inspired Intelligence for Real-Time Health Situation Understanding in Smart e-Health Home Applications. IEEE Access, 7, 180106-180126.

[8] Zhu, T., Colopy, G.W., MacEwen, C., Niehaus, K.E., Yang, Y., Pugh, C.W., & Clifton, D.A. (2019). Patient-Specific Physiological Monitoring and Prediction Using Structured Gaussian Processes. IEEE Access, 7, 58094-58103.

[9] Leu, F., Ko, C., You, I., Choo, K.R., & Ho, C. (2017). A smart phone-based wearable sensors for monitoring real-time physiological data. Computers & Electrical Engineering, 65, 376-392.

[10] Bhatia, M., & Sood, S.K. (2018). Exploring Temporal Analytics in Fog-Cloud Architecture for Smart Office HealthCare. Mobile Networks and Applications, 1-19.

[11] J.Li, J. Jin, J. D. Yuan, H. Zhang (2018). Virtual Fog: A Virtualization Enabled Fog Computing Framework for Internet of Things. IEEE Internet of Things Journal, vol. 5, pp. 121-131 [12] P. Wu, E.W. Ngai, Y. Wu, (2018). Toward a real-time and budget aware task package allocation in spatial crowdsourcing, Decision Support Systems, vol. 110, pp. 107-117.

[13] P.G.V. Naranjo, M. Shojafar, H. Mostafaei, Z. Pooranian, E. Baccarelli (2016). P-SEP: A prolong stable election routing algorithm for

energy limited heterogeneous fog-supported wireless sensor networks. The Journal of Supercomputing, vol. 73, no.2, 733–755
[14] V. Moysiadis, S. Panagiotis, I. Moscholios, (2018). Towards Distributed Data Management in Fog Computing, Wireless Communications and Mobile Computing, vol.2018, pp.1-14
[15] M. Lavassani, S. Forsstrom, U. Jennehag, T. Zhag, (2018). Combining Fog Computing with Sensor Mote Machine Learning for Industrial IoT, Sensors, vol. 18, no. 1532, 2018 [16] G. Peralta, P. Garrido, J. Bilbao, R. Aguero, P.M. Crespo, (2019). On the Combination of Multi-Cloud and Network Coding for Cost-Efficient Storage in Industrial Applications, Sensors, vol. 19, no. 7, pp. 1-19
[17] Y. Yu, L. Xue, Y. Li, X. Du, M. Guizani, B. Yang, (2018). Assured Data Deletion with Fine Grained Access Control for Fog-based Industrial Applications, IEEE Transactions on Industrial Informatics, vol. 14, no. 10, pp. 4538-4547, 2018
[18] F.H. Tseng, M.S. Tsai, C.W. Tseng, Y.T. Yang, C.C. Liu, L.D. Chou, "A Lightweight Auto-Scaling Mechanism for Fog Computing in Industrial Applications", IEEE Transactions on Industrial Informatics, vol.14, no.10, 4529-4537, 2018
[19] Y. Liu, K.A. Hassan, M. Karlsson, O. Weister, S. Ghong, "Active Plant Wall for Green Indoor Climate based on Cloud and Internet of Things", IEEE Access, vol. 6, pp. 33631-33644, 2018
[20] Neamatollahi, P., Naghibzadeh, M., & Abrishami, S. (2017). Fuzzy-based Clustering-Task Scheduling for Lifetime Enhancement in Wireless Sensor Networks. IEEE Sensors Journal, 17, 6837-6844.
[21] Li, J., Hou, X., Su, D., & Munyemana, JDD. (2017). Fuzzy overoptimized clustering routing algorithm for wireless sensor networks. IET Wireless Sensor Systems, 7(5), 130–137.
[22] Cacciagrano, D., Culmone, R., Micheletti, M., & Mostarda, L. (2019). Energy-Efficient Clustering for Wireless Sensor Devices in Internet of Things. Performability in Internet of Things, 59-80.
[23] Reddy, M.P., & Babu, M.R. (2017). A hybrid cluster head selection model for Internet of Things. Cluster Computing, 1-13
[24] Halder, S., Ghosal, A., & Conti, M. (2018). LiMCA: an optimal clustering algorithm for lifetime maximization of internet of things. Wireless Networks, 1-19.
[25] Li, G., Wu, J., Li, J., Wang, K., & Ye, T. (2018). Service Popularity based Smart Resources Partitioning for Fog Computing-enabled Industrial Internet of Things. IEEE Transactions on Industrial Informatics, 1–1.
[26] Shukla, S., Hassan, M. F., Jung, L. T., & Awang, A. (2018). Architecture for Latency Reduction in Healthcare Internet-of-Things Using Reinforcement Learning and Fuzzy Based Fog Computing. Recent Trends in Data Science and Soft Computing, 372–383.
[27] Miao, D., Liu, L., Xu, R., Panneerselvam, J., Wu, Y., & Xu, W. (2018). An Efficient Indexing Model for the Fog Layer of Industrial Internet of Things. IEEE Transactions on Industrial Informatics, 1–1.
[28] Shen, J., Yang, H., Wang, A., Zhou, T., & Wang, C. (2018). Lightweight authentication and matrix-based key agreement scheme for healthcare in fog computing. Peer-to-Peer Networking and Applications.
[29] K. Bashir Shaban, A. Kadri, E. Rezk (2016). Urban Air Pollution Monitoring System with Forecasting Models, IEEE Sensors Journal, vol. 16, no. 8, pp. 2598–2606.
[30] K. Hu, A. Rahman, H. Bhrugubanda, V. Sivaraman (2017). HazeEst: Machine Learning Based Metropolitan Air Pollution Estimation from Fixed and Mobile Sensors. IEEE Sensors Journal, vol. 17, no. 11, pp. 3517–3525.
[31] S. Beulah, F. R. Dhanaseelan, (2018). An Optimal Method for Duplication Detection and Elimination from Air Pollution Data of Wireless Sensor Network, International Journal of Environment and Waste Management, vol.21, no. 2/3.
[32] S. Dhingra, R. B. Madda, A. H. Gandomi, R. Patan, M. Daneshmand, M. (2019). Internet of Things Mobile - Air Pollution Monitoring System (IoT-Mobair), IEEE Internet of Things Journal, 1–1.
[33] Z. Hu, Z. Bai, Y. Yang, Z. Zheng, K. Bian, Song, L. (2019). UAV Aided Aerial-Ground IoT for Air Quality Sensing in Smart City: Architecture, Technologies, and Implementation, IEEE Network, vol. 33, no. 2, pp. 14–22
[34] L. Luo, Y. Zhang, B. Pearson, Z. Ling, H. Yu, X. Fu, (2018). On the Security and Data Integrity of Low-Cost Sensor Networks for Air Quality Monitoring. Sensors, vol. 18, no. 12, 4451.
[35] Adhikari, M., Mukherjee, M., & Srirama, S.N. (2019). DPTO: A Deadline and Priority-aware Task Offloading in Fog Computing Framework Leveraging Multi-Level Feedback Queueing. Internet of Things Journal.
[36] Adhikari, M., & Gianey, H.K. (2019). Energy efficient offloading strategy in fog-cloud environment for IoT applications. Internet of Things, 6, 100053.
[37] Ni, J., Zhang, K., Yu, Y., Lin, X., & Shen, X. S. (2018). Providing Task Allocation and Secure Deduplication for Mobile Crowdsensing via Fog Computing. IEEE Transactions on Dependable and Secure Computing, 1–1.
[38] Borujeni, E. M., Rahbari, D., & Nickray, M. (2018). Fog-based energy efficient routing protocol for wireless sensor networks. The Journal of Supercomputing.
[39] Liu, Z., & Li, S. (2018). Sensor-cloud data acquisition based on fog computation and adaptive block compressed sensing. International www.phdprime.com 22 22 Journal of Distributed Sensor Networks, vol. 14. Issue. 9, 155014771880225.
[40] Fu, J., Liu, Y., Chao, H.-C., Bhargava, B., & Zhang, Z. (2018). Secure Data Storage and Searching for Industrial IoT by Integrating Fog Computing and Cloud Computing. IEEE Transactions on Industrial Informatics, 1–1.
[41] Bathiya, B., Srivastava, S., & Mishra, B. (2016). Air pollution monitoring using wireless sensor network. 2016 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)