Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

SmartGuard: An Improved Leaky Bucket Algorithm-Based Secure and Scalable Congestion Control Framework for the Edge-Cloud Continuum

Frimpong Twum [1], John Kwao Dawson [1], Ben Beklisi Kwame Ayawli [2], Bright Bediako Kyeremeh [2] and Solomon Antwi Buabeng [2]

[1] *Kwame Nkrumah University of Science and Technology, Department of Computer Science, Ghana*
[2] *Sunyani Technical University, Department of Computer Science, Ghana*
* *Correspondence: author: Dr. John Kwao Dawson (e-mail: kwaodawson@stu.edu.gh).*

ARTICLE INFO

Received: 26 Dec 2024

Revised: 14 Feb 2025

Accepted: 22 Feb 2025

ABSTRACT

Edge computing has drawn a lot of attention as a potential cloud computing revolution recently. However, the edge will almost always be used in conjunction with cloud computing. Edge computing is therefore a subset of cloud computing. Large amount of data have been collected over time using previously unheard-of computer power and linked devices, putting enormous strain on the already overburdened internet. This massive data collection creates issues with latency and put enormous constraints on bandwidth. The sensors used to send real time data to the cloud are limited in providing security for the motion data, and creates other issues such as delay packet delivery, drop packets and congestion. This study proposed a Hybrid Leaky Bucket Algorithm (HLBA) which is an integration of Pascal's Triangle, Kadane's Algorithm and Traditional Leaky Bucket Algorithm to secure packet during transfer, eliminating delay in packet delivery and avoiding congestion. The algorithm was implemented using python on CloudSim and tested using packet sizes of 5 bytes, 10 bytes, 30 bytes, 70 bytes and 100 bytes using artificially generated packets. From the study, with a LeakRate of 2 and capacity of 2, the proposed algorithm used 0.00034 milliseconds to transfer a packet length of 5 bytes of data. This gives a 99.257% difference in packet transfer time between the proposed HLBA and conventional Leaky Bucket algorithm.

Keywords: Kadane's Algorithm; Leaky Bucket Algorithm; Pascal's Triangle; Sensors; Traffic Congestion

1. INTRODUCTION

Edge computing has received a lot of attention lately as a possible revolution in the cloud computing space. Cloud computing will, nevertheless, almost always be utilized in conjunction with edge computing. This makes edge computing a subset of cloud computing. Edge computing involves hosting apps and data closer to end users, either on the customer's premises or in smaller edge data centers, while cloud computing involves hosting apps and data in a core data center [1]. Large volumes of data have been gathered over time by previous unheard-of amount of computer

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

power and linked devices, placing an enormous burden on the already overloaded internet. This enormous data collection causes problems with latency and bandwidth [2]. Edge computing provides a better option to shift away from handling complicated data to close vicinity to the original data source, in contrast to typical corporate computing, where data is generated at the client's end or the user's computer. Edge computing becomes the solution that every modern organization needs, from less internet traffic to fewer latency problems, quicker reaction times, lower security risks, better-performing applications, deeper insights, and crucial data analysis to enhanced customer experiences [3]. The transfer of analyzed data from edge nodes to the cloud has resulted in a growing number of internet services being built of which two well-known ones are Akamai's Content Delivery Network (CDN) and Amazon's Elastic Compute Cloud (EC2) [4]. With the help of these cloud-based services, customers may get the features they need without having to setup and operate the services locally or have a thorough grasp of the underlying architecture. Cloud-edge based services are a major trend in contemporary technology since they provide several benefits, such as scalability, affordability, cheap storage and accessibility [5]. Due to these benefits, the embrace of edge computing has quickened. Wireless Sensor Networks (WSN) are used as congestion control approaches in the transfer of real time data from edge computing to the cloud infrastructure as depicted in Figure 1. WSN are used in a number of other fields, including transportation, agriculture, real-time security, intelligence gathering, e-healthcare to gather huge amount of data [6]. The use of WSN in the transfer of data from edge computing to the cloud is determined in terms of small size, low cost, and power consumption using a large number of sensor nodes. However these WSN are limited in providing security for the motion data from the edge to the cloud. This make the transmission process vulnerable. This raises the need to propose a Hybrid leaky Bucket algorithm as a congestion control mechanism to regulate traffic congestion and also secure the transferred data in a cloud-edge ecosystem.

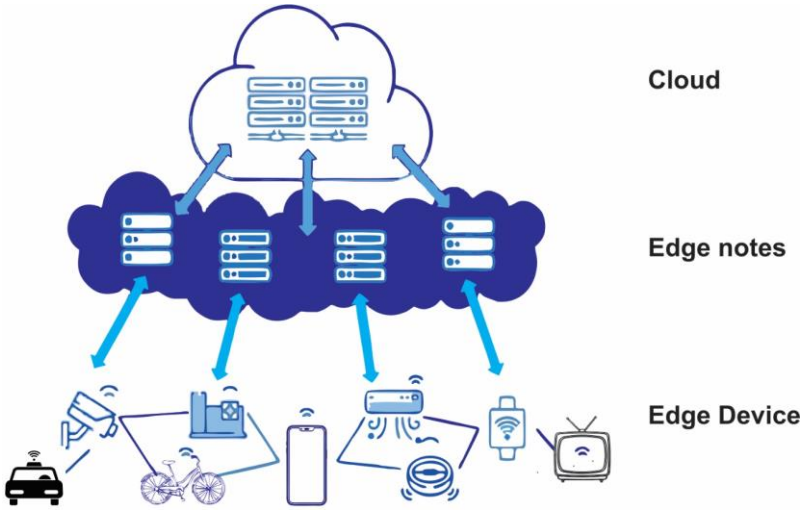

Figure 1: Transfer of data from edge computing to cloud storage.

1.1. Leaky Bucket Algorithm For Packet Transfer

The primary goal of the leaky bucket algorithm, a popular technique for traffic shaping and rate limiting in networks is to achieve smooth burst traffic and restrict the network's data transmission rate as indicated in Figure 3.Traffic bursts with irregular input rates are converted into a regular stream of equal-gap data by the leaky bucket rate-limiting algorithm. The output traffic rate is fixed, independent of the size of the incoming traffic [28]. The infrequent data stream is sent by the leaky bucket algorithm to a packet queue that is managed by a leaky bucket queue controller. The bucket is invalid when the packet arrival rate is less than the bucket's output rate, likewise when the bucket's arrival rate exceeds its output rate as shown in Figure 2. The infrequent data stream is sent by the leaky bucket algorithm to a packet queue that is managed by a leaky bucket queue controller. After receiving the packets to be

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

forwarded, the queue feeds them into the leaky bucket while it awaits scheduling to determine the forwarding timing. The appropriate forwarding action is carried out after comparing the packet input rate into the bucket with the input rate at the bottom of the bucket [28]. The bucket is invalid once the packet arrival rate is less than the bucket's output rate which can be computed using equation 1. Also there is an error when the bucket's arrival rate exceeds its output rate. Congestion controls are to smoothen out data transfer, however, due to limited bandwidth resources, network congestion is likely to occur in traditional leaky bucket if the incoming rate reaches a peak.

This study therefore proposes a hybrid Leaky Bucket algorithm that integrates Pascal's Triangle, Kadane's algorithm with traditional Leaky Bucket algorithm to smoothen data transfer, reduce network congestion, and provide security for tranfered packets. In the proposed algorithm, the contagious sum from the Kadane's algorithm is used as the queue controller value. So there will be no readjustment of queue controller value for packets with size more than the capacity of the leaky bucket as depicted in Figure 5 and 6 and algorithm 1.

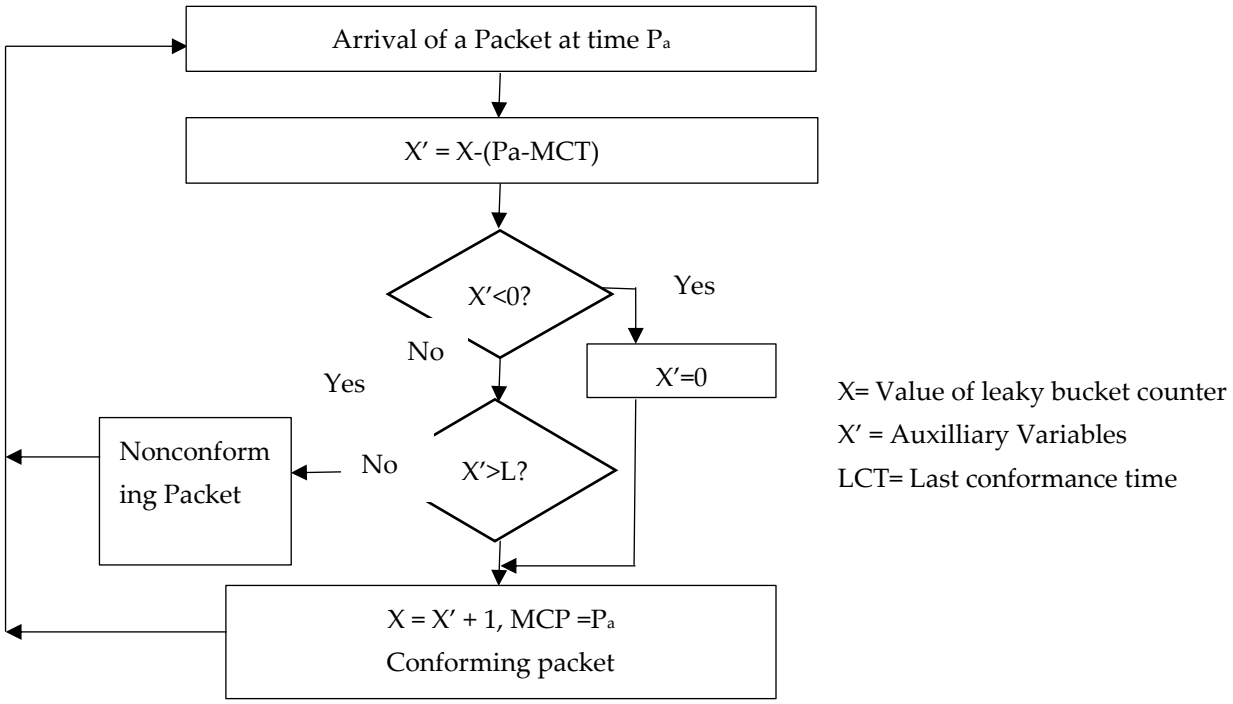

Figure 2: Flowchart for Conventional Leaky Bucket Algorithm

$$X(t) = \min(B, X(t-1) + A(t) - R) \quad (1)$$

Where; B is considered the size of the bucket in packet or bytes

R is the fixed LeakRate (byte per seconds)

$A(t)$ the incoming packet rate at time t

$X(t)$ the amount of packets at time t

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

There is always overflow when $X(t) > B$. This causes the loss of packets. Again, when $A(t) > R$, packets are buffered which causes delay in packet transmission.

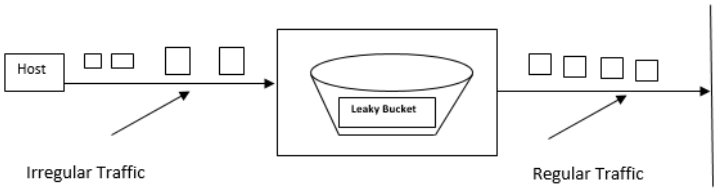

Figure 3: Conventional Leaky bucket algorithm data flow

1.3. Identified Problem

The volume of data grows proportionally with the number of devices (sensors) linked to edge computers.If these processed data are not required right away, they are then sent to the cloud. To prevent packet overflow during transmission has necessitated a congestion control strategy. The goal of a congestion control mechanism is to distribute a network's resources such that, in situations where demand exceeds or approaches the network's capabilities, the network can function at a satisfactory level. These resources consist of buffer space (memory), link bandwidths, and processing power at intermediary nodes [7], [8]. Studies have shown that, the best approach is through the use of leaky bucket algorithms [7]. The leaky bucket algorithm, as a congestion control mechanism consequences include high traffic and significant data leakage in the network resulting in the inability of the sensor nodes forwarding the data quickly during the data transmission process which leads to data loss and data leakages [9], [10], [11]. As the load increases, the problem becomes more significant because low overhead and equality become more crucial. Under high load, the bandwidth (or channel capacity) might well be significantly decreased in the absence of appropriate congestion control methods and this will impact on throughput [12]. Again higher data security is also achieved when edge computing is housed on client's property as opposed to the cloud [12]. Potential hackers are able to intercept data sent from the edge to the cloud [13]. Therefore, there is the need for a more robust security congestion control approach capable of reducing congestion and also providing security. Hence, this study integrates Pascal's triangle, Kadane's algorithm and leaky bucket algorithm in a proposed Hybrid Leaky Bucket Algorithm (HLBA) as an improved solution for securing packet during transmission, eliminating delay in packet delivery and avoiding network congestion in edge-cloud continuum.

2. RELATED WORKS

Research on algorithmic congestion control in edge-cloud consortium solutions is now very advanced, with many researchers focusing on improving or simultaneously optimizing metrics like energy usage and time delay.The majority of offloading frameworks are based on mathematical models, whereas the majority of computational offloading strategies are based on intelligent optimization algorithms like whale optimization and genetic algorithms. Collaborative edge –cloud computing offloading techniques have also been researched [14].Task upload coordination

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

and task migration between Mobile Edge computing (MEC) /cloud servers, and their devise computation capabilities of all thoroughly examined in the study of Xu et al., [15]. Their algorithm was very promising but lacked the ability to secure packets in a cloud-edge continuum.

A processing model for the issue of balancing energy consumption and processing load-balancing in edge computing is presented in the work of Abbasi et al., [16]. Based on this, they designed a cooperative computation offloading (CCO) scheme by using queuing theory to describe the task upload, migration, and computing process with the goal of reducing task completion latency. The NSGAII algorithm is used to solve the problem of multi-objective model. Even though their algorithm was promising exact percentage of delay reduction and energy consumption is not stated likewise lack security properties.The collaborative computation offloading problem in a heterogeneous edge-cloud system with different users was investigated by Ahmed et al., [17]. An energy consumption minimization approach with combined bandwidth and computational resources allocation was presented based on dynamic planning in their work. The simulation results demonstrated a decrease in mobile device energy consumption but did not consider delay transmission of data as well as security of transmitted packets. An energy-efficient and delay-aware offloading strategy (EEDOS) based on Device to Device (D2D) cooperation in MEC was proposed in the work of Ranji et al., [18]. Their work even though reduced packet delay lacked the ability to secure the packets. In the work of Asghari et al., [19], a novel Energy-aware server placement technique that used a Dynamic Voltage and Frequency Scaling (DVFS) technique and tree social relation algorithm (ESPT) to place edge servers optimally in order to increase network coverage was proposed. By their approach a Mobile device can delegate their task to the MEC or another nearby idle mobile device in EEDOS but lacked security features. Mondal et al., [20] suggested a Dynamic Energy-Efficient Offloading Algorithm (DEEO) as a solution to the problem of congestion control in an edge-cloud settings. DEEO gives mobile devices the ability to effectively transfer computationally demanding secure picture transmission duties to the closest fog access point or edge server to the cloud. The Heterogeneous Distributed Deep Learning with Data Offloading (DDLO) algorithm was also proposed by Yang [21] to augment the work of Mondal et al. DDLO improved data processing by utilizing the synergy between clouds' computing and edge computing but lacked security properties. An innovative approach that used an ant colony optimization to prioritize several scheduling objectives and calculate a fitness function was presented in the work of Khaleel et al., [22]. To improve computational efficiency, the algorithm efficiently decides how to divide up applications across edge and cloud servers. Despite ensuring speedy and successful data transfer, their method is devoid of security safeguards. Hosny et al., [23] suggested an Enhanced Whale Optimization Algorithm (EWOA) to optimize dependent task offloading in an edge-cloud environment. By minimizing overall processing delay, energy consumption, and related expenses, their suggested method sought to determine the best offloading situation for dependent activities but lack security properties. From the reviewed literature, all the proposed approaches were able to reduce delays and loss of packets during transmission between edge-cloud environments but could not ensure the security of the packets in motion

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

3. METHODOLOGY

In this study, a Hybrid Leaky Bucket Algorithm (HLBA) was proposed to ensure security and congestion free environment in the transfer of packets collected by wireless network sensors from edge devices through edge nodes to the cloud. The Hybrid leaky bucket algorithm is an integration of Pascal's triangle [24], Kadane's algorithm [25] and Conventional leaky bucket algorithm [26]. The Hybrid leaky bucket algorithm is faced into three levels as indicated in Figure 4 as against the conventional Leaky Bucket flowchat in Figure 2.The data generated by the wireless network sensors are imperiled to the Pascal's triangle formula which encrypts the packets at the first level. The encrypted packets are then subjected to the Kadane's algorithm to select the maximum value as the initialization counter for the leaky bucket queue controller . There is no chance of any undelivered packets in the network because the data packets are sent in accordance with the maximum value from Kadane's algorithm, which is used as the base value for the networks total packet size. Packet buffering only occurs at the edge router. Thus, it can be applied to guarantee equitable bandwidth distribution and improve transient performance.

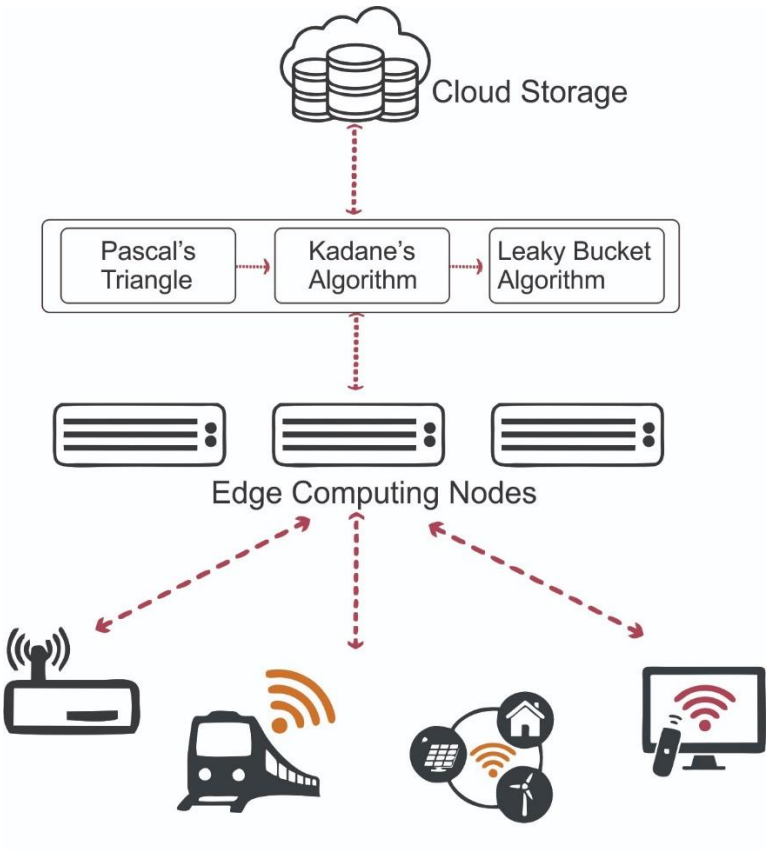

Figure 4. Flow diagram for proposed security and congestion control approach.

3.1. Proposed Algorithm

The proposed Hybrid leaky bucket algorithm integrates Pascal's Triangle, Kadane's algorithm and Conventional Leaky bucket algorithm to secure and reduce congestions in the transfer of data in an edge-cloud continuum is indicated in Figure 4, with the corresponding encryption and decryption flowchart depicted in Figure 5 and 6. The

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

proposed scheme has the advantage of reducing data loss, data drop and runtime when the bucket's arrival rate exceeds its output rate [29]. In the proposed algorithm, the Pascal's Triangle is applied on the packet to be transmitted. The Pascal's Triangle, scrambles the packet to produce a Ciphertext. The Kadane's algorithm is then applied on the Ciphertext to compute the contagious maximum sum value to be used as the initialization value for the leaky bucket queue controller. The computation of the maximum value helps reduce the shortfall of the leaky bucket, readjusting to select a new initialization value when the packet size is higher than the size of the queue controller remaining value.

3.2. Pascal's Triangle for Encryption

The Pascal Triangle is named after a French mathematician and scientist Blaise Pascal. This mathematical formula provides a rich supply of mathematical features [27]. The Pascal's Triangle uses a binomial coefficient formula as indicated in equation 2, which is used to encrypt the packet to be transmitted from the edge to the cloud.

$$T(n,k) = (n,k) = \frac{n!}{k!(n-k)!} \qquad (2)$$

7. The Explanation of This Theory; $T(n,k)$ indicates the value for the elements in the $n-th$ row and the $k-th$ column (all starting from 0) $n!$ Is the factorial of n, which is defined as $n*(n-1)*(n-2)---*1$ $k!$ and $(n-k)!$ are the factorials for k and $(n--k)$, respectively.

3.3 Key Details

The first row $(\boldsymbol{n-0})$ contains one element, $\boldsymbol{T(0,0)=1}$. The number of $\boldsymbol{T(n,k)}$ is the total of the elements directly above it in the Pascal's Triangle using equation 3.

$$\boldsymbol{T(n,k) = T(n-1,k-1) + T(n-1,k)} \qquad (3)$$

With $\boldsymbol{T(n,k)=0}$, when $\boldsymbol{k<0}$ or $\boldsymbol{k>n}$

For example;

Considering the $\boldsymbol{4th}$ row of Pascal's Triangle, the following will be the output; $\boldsymbol{T(4,0)=(4,0)=1, T(4,1)=(4,1)=6, T(4,3)=(4,3)=4, T(4,4)=(4,4)=1}$

This results in $[1,4,6,4,1]$ for the $4th$ row.

3.4 . Decryption Function for Pascal's Triangle

The reversal function for the Pascal's Triangle is applied in the cloud to decrypt the stored packet to be made available in the edge for usage by cloud clients. The formula for the reversal fucntion is indicated in equation 4.

$$\mathrm{T(n,k)= (n,k)= } \frac{n!}{k!(n-k)!} \qquad (4)$$

The value for n and k are now computed. To reverse the pascal's function, the n and k are determinedusing equation 5, such that;

$$T(n,k) = X \qquad (5)$$

From this, X is the given value for the Pascal's Triangle. Hence in computing n, the sum of all the elements up to the $n-th$ row is given as;

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$S_n = 2^n$ (6)

By computing the smallest n such that $S_n \geq X$, we know which row n the value X belongs to. In the next round, the value for k is computed. Since n is already computed, iterate over $k = 0$ to n, using equation 7;

$T\ (n,k) = (n,k)$ (7)

Evaluate for which $k, T(n,k) = X$ using equation 8. This results in the column position.

$$D[i] = \frac{E[i]}{PascalWeight\ [i \bmod 5]} \quad (8)$$

3.5 Kadane's Algorithm to Compute Contagious Maximum Sum

Kadane's algorithm is a scheming solution to the O(n) problem. The concept of dynamic programming forms the basis of the fundamental premise for Kadane's algorithm [27]. In computing Kadane's contagious maximum sum, assume M_i as the maximum contagious sum ending at and including index i *for* $\mathbf{0 \leq i \leq n-1}$. Then we have $\boldsymbol{M_0 = A[0]}$ and $\boldsymbol{M_i = \max\ (M_{i-1} + A[i], A[i])}$, where A is the array of numbers from the Pascal's Triangle. This indicates that $\boldsymbol{M_0 = A[0]}$, since $\boldsymbol{A[0]}$ is the exact contagious with an ending index $\mathbf{0}$. For example, $\boldsymbol{M_i}$ for $\mathbf{1 \leq i \leq n-1}$ Where $\boldsymbol{C_i}$ is considered to be the contagious sum at index $\boldsymbol{i}$, denoted as $\boldsymbol{C_i - 1}$ ending at index $\boldsymbol{i-1}$. From this, the maximum contagious sum is computed using equations 9, 10, 11 and 12.

$$C_i = \{x + A[i] | x \in C_{i-1} \cup \{A[i]\} \quad (9)$$

$$M_i = \max(C_i) = \max(\{x + A[i], \in C_i - 1\} \cup \{A[i]\}) \quad (10)$$

$$= \max\ (\{x + A[i] x, \in (i-1)\}\ , A[i]) \quad (11)$$

$$= \max\ (Mi - 1 + A[i], A[i]) \quad (12)$$

From equation 9, 10, 11 and 12, calculating the maximum contagious sum could be done using the maximum contagious sum which ends at a particular value from the index $i = 1, .., n-1$ based on the initial contagious high value ending and including index $i - 1$.

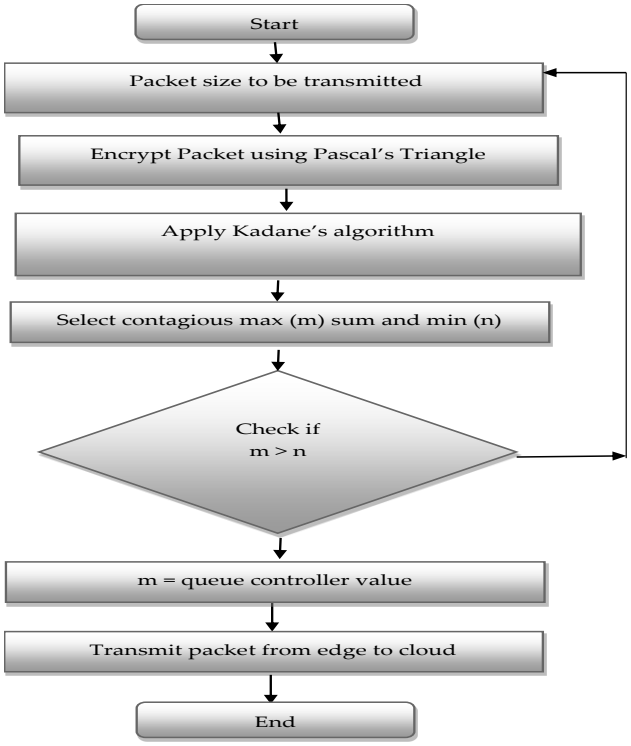

Figure 5. Flowchart for encryption of packet for the proposed algorithm.

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

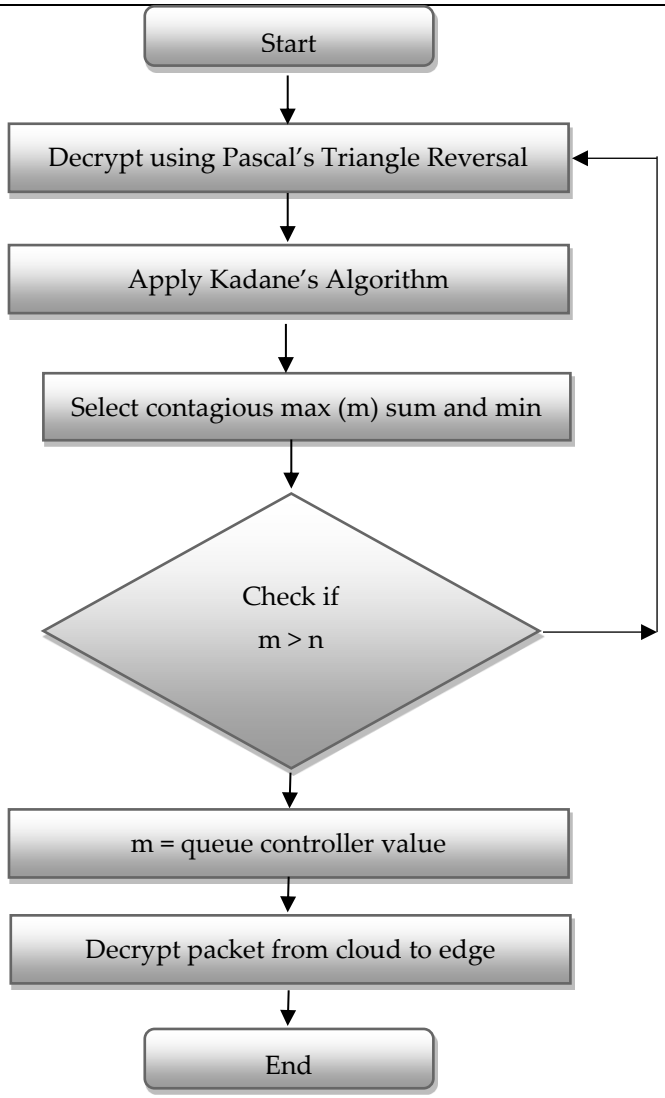

Figure 6. Flowchart for decryption of packet for the proposed algorithm.

From Figures 5 and 6, the drop and delay in packet transmission is solved by using the contagious sum from the Kadane's algorithm as the value for the queue controller. Therefore from equation 12, B is always greater than $X(t)$ and also R is always greater than $A(t)$. This is occasioned by the contagious sum from the Kadane's algorithm.

Algorithm 1: PROPOSED ALGORITHM FOR HIETHIENED LEAKY BUCKET ALGORITHM

Algorithm 1: Hybrid Leaky Bucket

Input: Packet Stream $P[i]$ arriving

Output: Secure and Congestion-Controlled Packet transfer to the Cloud

1: $W[i] =$ PascalWeight $[i \ mod \ 5]$ //compute Pascal's Triangle weight

2: $E[i] = P[i] * W[i]$ //encrypted data

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

3: maxSum $= E[0], CurrentSum = E[0]$ //initialization variable

4:For $j = 1$ to $n - 1$ {

5: CurrentSum= max ($[E[j], CurrentSum + E[i]$)

6: maxSum= max($maxSum, CurrentSum$)

7: }

8: $C = maxSum$ //Kadane's result used as Leaky Bucket Capacity to prevent congestion

9: $L = LeakRate, Level = 0$ //initialize leaky bucket

10: Level= max $(0, Level - L * (timeElapsed))$

11: if $Level + p[i] \leq C$ {

12: Accept Packet and add to bucket

13: }

14: else {

15: Drop packet due to congestion

16: }

17: $W[i] = Pascal'sWeight[i \ mod \ 5]$// compute decryption weight

18: $D[i] = \frac{E[i]}{W[i]}$ //Decrypt Packet

19: Transmit decrypted packet from the cloud to edge

4. ENVIRONMENTAL SETUP

The simulation was run on an i7 Lenovo computer which has a processing specification of 2.10 GHz and implemented on CloudSim using python programming language. Personal varying packet sizes were used to test the robustness of the proposed scheme. From this, a packet size of $P = [3, 5, 7, 2, 6, 4]$, was used as illustrated in example 1. The experiment used $TimeStamp$ with single second interval at a $LeakRate$ of 2.

4.1 Experimental Results

Example 1: The Packets to be transferred to the cloud from edge computing is as indicated in equation 13;

$$\boldsymbol{P} = [\mathbf{3}, \mathbf{5}, \mathbf{7}, \mathbf{2}, \mathbf{6}, \mathbf{4}] \quad (13)$$

TimeStamp to be used by leaky bucket is as indicated in equation 14

$$T = [0,1,2,3,4,5 \ldots] Using \ a \ single \ second \ interval \quad (14)$$

LeakRate:

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$$L = 2$$

The Rate of Execution:

Using Pascal's Triangle Encryption: The packets to be transferred are encrypted using equation 15.

$$E = [3 * 1, 5 * 4, 7 * 6, 2 * 4, 6 * 1, 4 * 4] = [3,20,42,8,6,16] \quad (15)$$

Apply Kadane's Algorithm: Compute the contagious sum using equation 16.

$$MAXsum = MAX([3], [3 + 20], [3 + 20 + 42], [3 + 20 + 42 + 8 + 16) = 89 \quad (16)$$

$$\text{Queue Controller Capacity } = C = 89$$

Decryption from cloud to edge nodes; apply the reversal of the Pascal's function to decrypt packets from the cloud to the edge notes using equation 17 with the output depicted in Figure 7.

$$\mathrm{D[i]} = \frac{\mathrm{E[i]}}{\mathrm{PascalWeight\ [i\ mod\ 5]}} \quad (17)$$

$$D = \left[\frac{3}{1}, \frac{20}{4}, \frac{42}{6}, \frac{8}{4}, \frac{6}{1}, \frac{16}{4}\right] = [3,5,7,2,6,4]$$

4.2 Simulation Results:

Beginning with $C = 89, L = 2, Level = 0$

```
Packet 3 (encrypted: 3) accepted. Bucket level: 3
Packet 5 (encrypted: 20) accepted. Bucket level: 23
Packet 7 (encrypted: 42) accepted. Bucket level: 65
Packet 2 (encrypted: 8) accepted. Bucket level: 73
Packet 6 (encrypted: 6) accepted. Bucket level: 79
Packet 4 (encrypted: 16) accepted. Bucket level: 85
```

Figure 7. Results of simulation for Hybrid Leaky Bucket Algorithm.

14. Results and Discussion

This section details out the results from the simulation of the algorithm based on packet length of 5 bytes, 10 bytes, 30 bytes, 70 bytes and 100 bytes. Figures 8, 9, 10 and 11 represents the simulation results from the experiment conducted.

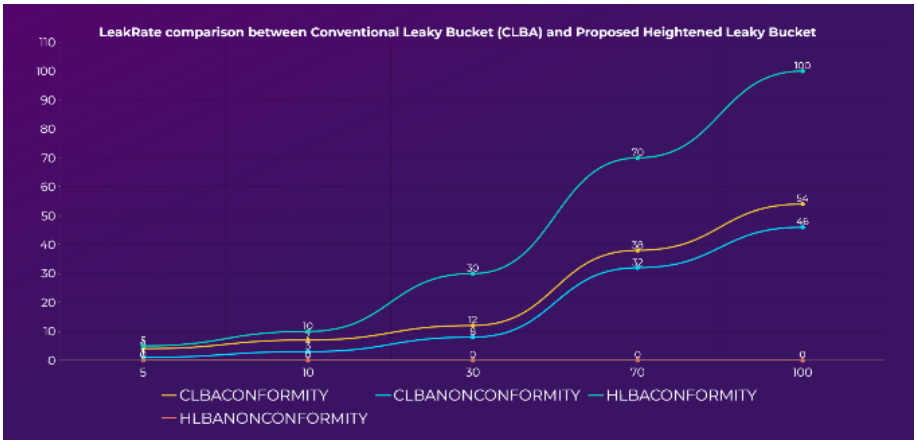

Figure 8. LeakRate comparison between Conventional Leaky Bucket (CLBA) and Proposed Hybrid Leaky Bucket.

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

From Figure 8, with a packet length of 5 bytes and a LeakRate of 2 packet per milliseconds, the conventional Leaky Bucket had 4 bytes of packets conforming to the set limits with 1byte of packet delaying for renewal of queue controller value. When the packet length was increased to 70 bytes, 38 bytes of packets conformed to the set limit with 32 bytes of packets delaying for queue controller value renewal. However, from Figure 8, with packet length of 5 bytes, there was a whole packet conformity. Again when the packet length was increased to 70 bytes, all the packets were seamlessly transferred because the queue controller value after transferring the 5 bytes length (600 packets) was still more than the 70 packet length (530 packets).

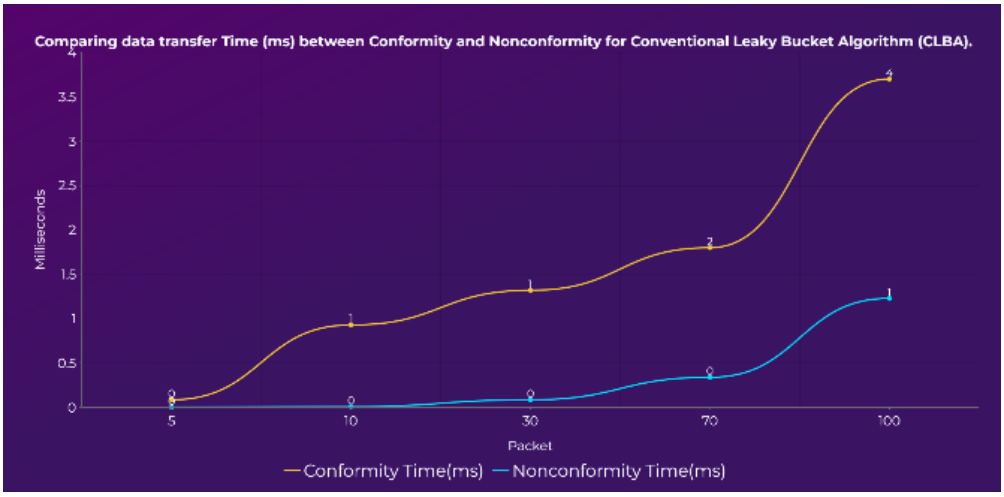

Figure 9. Comparing data transfer Time (ms) between Conformity and Nonconformity for Conventional Leaky Bucket Algorithm (CLBA).

From Figure 9, it took 0.087 milliseconds for a packet length of 5 bytes to be transmitted with a delayed time of 0.0037 milliseconds for renewal of queue controller value. Again when the packet length was increased to 70 bytes, 1.8 milliseconds was used in transferring the packet to the cloud from the edge with a corresponding 0.34 milliseconds waiting time.

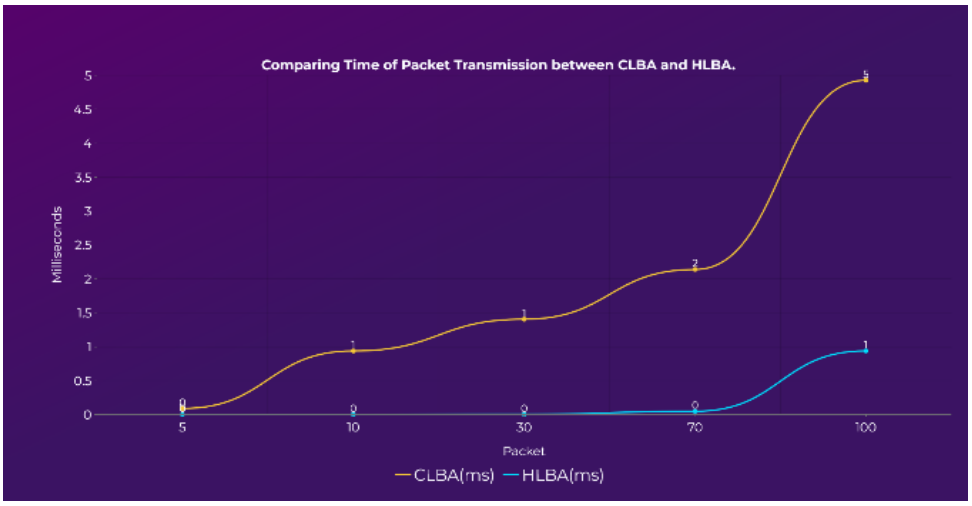

Figure 10. Comparing Time of Packet Transmission between CLBA and HLBA.

From Figure 10, with a packet length of 5 bytes, 0.0907 milliseconds was used to transfer both conforming and nonconforming packets to the cloud from edge computing devices under the CLBA while 0.00034 milliseconds was used to transfer the full packet length of 5 bytes from the edge nodes to the cloud under the HLBA. Again, 4.93

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

milliseconds was used to transfer full (conforming and nonconforming) packets size of 100 bytes to the cloud from the edge devices while a 0.938 milliseconds was used to transfer a packet with length 100 bytes using the HLBA. This seamless transmission of packets using the HLBA resulted from the application of the Kadane's algorithm to select the maximum value from the Pascal's Triangle serving as the value for the queue controller value.

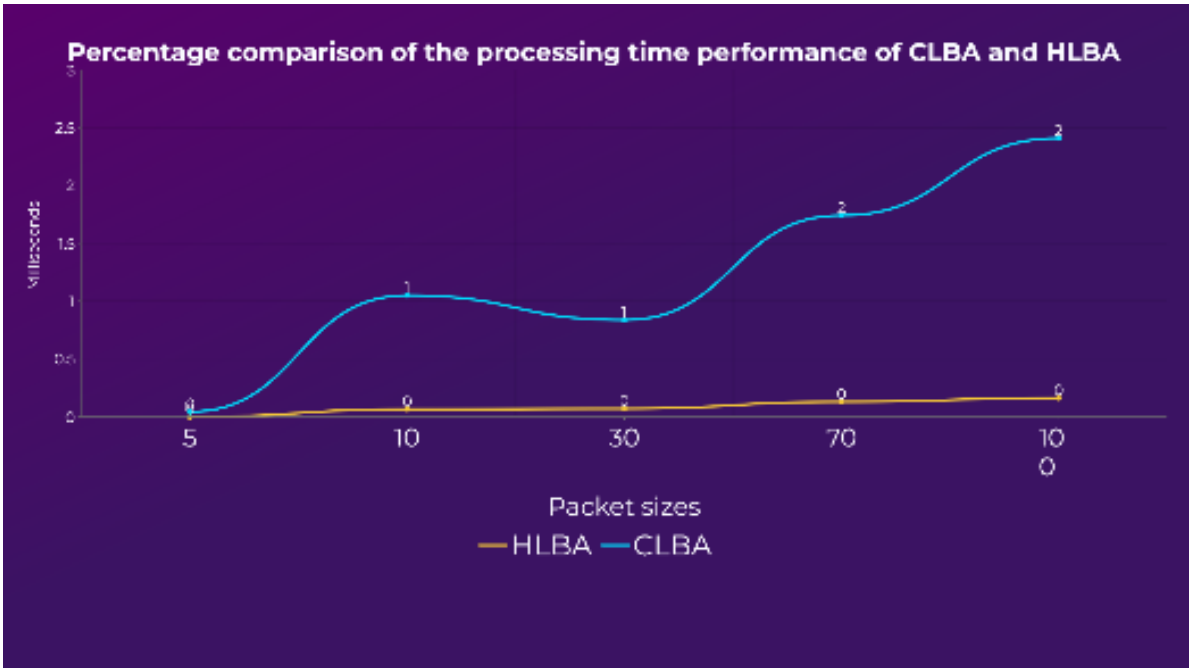

Figure11: Percentage comparison of the processing time performance of CLBA and HLBA

From Figure 11, an average of 30 runtimes was conducted and their percentage improvement considered. From Figure 11, there was a percentage improvement of 99.257% when 5 bytes of packets was transmitted using the HLBA as compared with the CLBA. Also, with a packet size of 10bytes, the HLBA had a 93.671% improved over transmission using CLBA. Again with a packet size of 100 bytes, the run time for CLBA was 2.409687 milliseconds while the HLBA had a run time of 0.16379 milliseconds. This gives a 93.203% improvement of the HLBA over the CLBA. From this discussion it could be concluded that the HLBA maintained a high performance across all packet sizes, with improvements ranging from 91.439% to 99.257%. Furthermore, because the transmission rate is high there was low power consumption since the processor consumption power was low [30].

4.3 TRANSFER TIME EVALUATION

During the packet transfer using a fixed bandwidth, all the packet sizes showed consistent bandwidth improvement. The HLBA used a bandwidth of 9280.00 bps for the transfer of the five (5) different packet sizes with the CLBA using a bandwidth of 8800.00bps. This gives a bandwidth percentage reduction of 5.17 between the bandwidth for HLBA and CLBA. The performance advantage remained for bigger packet sizes, with a 93.203% gain for 100-byte packets. From the transfer rates, with a data size of 5 packets, both the CLBA and HLBA used 5.00 milliseconds (converted to nearest whole number) to upload and download data from the cloud. When the data size was increased to 10 packets,

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

the HLBA used 9.00 milliseconds with the CLBA using 10 milliseconds. Also when the data size was increased to 100 packets, the HLBA used 90.00 milliseconds to upload and download data from the cloud as indicated in Figure 12. However, the CLBA used as high as 95.00 milliseconds to upload and download data from the cloud. From this it could be concluded that, the proposed HLBA uses less bandwidth to upload and download data to and from the cloud accounting from the seamless transmission of from the integration of Pascal's Triangle, Kadane's Algorithm and the Traditional Leaky Bucket Algorithm.

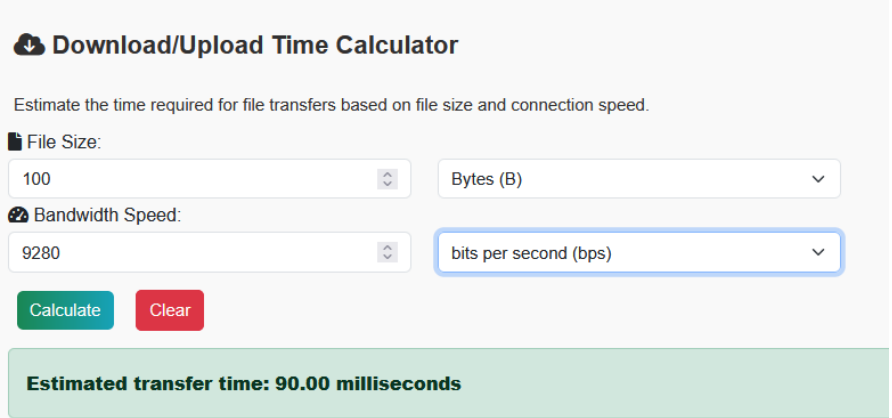

Figure 12: Data upload and download run times for different data sizes.

5. CONCLUSION

This study proposed a Hybrid Leaky Bucket algorithm (HLBA) to improve transferred packet's security, reduce congestion, packet delay and drop of packet during transmission in edge-cloud continuum. The HLBA improved the CLBA by integrating Pascal's Triangle for security and Kadane's algorithm for optimization. The implementation significantly improved processing time and bandwidth efficiency, proving its effectiveness for safe and efficient packet handling in network contexts. The proposed algorithm was able to secure packets and avoid packets delays thereby reducing traffic congestion at the edge-cloud continuum. Compared with conventional Leaky Bucket algorithm, the Hybrid Leaky Bucket Algorithm (HLBA) provided an improved results for secure packets transmission and congestion control. The HLBA reduces bandwidth usage by minimizing packet overhead and eliminating iterations for starting values, compared with Conventional Leaky Bucket algorithms. These results indicate HLBA's effectiveness in securing packet transmission, decreasing network resource utilization, eliminating packet delivery delays, and preventing congestion in edge computing situations.

6. FUTURE WORKS

Analyzing the suggested algorithm's security performance, a future work on its ability to resist man-in-the-middle attacks should be a focus. Again to help ensure seamless packet transfer in networks, further research should look into integrating the proposed framework with current traffic control systems such as ramp metering. Once more, to minimize packet transfer delays, the suggested framework would be used in real-time environment such as VANET, Smart Cities, Video Conferenceing and online gaming.

Funding: This study has no funding.

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

REFERENCES

[1]M. Ishtiaq, N. Saeed, and Muhammad Asif Khan, "Edge Computing in IoT: A 6G Perspective," *OPAL (Open@LaTrobe) (La Trobe University)*, Nov. 2021, doi: https://doi.org/10.36227/techrxiv.17031665.

[2]A. Aldoseri, K. N. A. - Khalifa, and A. M. Hamouda, "Re-Thinking Data Strategy and Integration for Artificial Intelligence: Concepts, Opportunities, and Challenges," *Applied Sciences*, vol. 13, no. 12, pp. 7082–7082, 2023, doi: https://doi.org/10.3390/app13127082.

[3] S. Liang, S. Jin, and Y. Chen, "A Review of Edge Computing Technology and Its Applications in Power Systems," *Energies*, vol. 17, no. 13, p. 3230, Jan. 2024, doi: https://doi.org/10.3390/en17133230.

[4]Y. Teng, J. Cui, and W. Jiang, "Research on Application of Edge Computing in Real-time Environmental Monitoring System," *Journal of Physics: Conference Series*, vol. 2010, no. 1, pp. 012157–012157, Sep. 2021, doi: https://doi.org/10.1088/1742-6596/2010/1/012157.

[5] L. Zhang, Y. Li, W. Yang, Q. Liu, and R. Yang, "PFTB: A Prediction-Based Fair Token Bucket Algorithm based on CRDT," May 2024, doi: https://doi.org/10.1109/cscwd61410.2024.10580549.

[6] X. Li, L. Zhu, X. Chu, and H. Fu, "Edge Computing-Enabled Wireless Sensor Networks for Multiple Data Collection Tasks in Smart Agriculture," *Journal of Sensors*, vol. 2020, pp. 1–9, Feb. 2020, doi: https://doi.org/10.1155/2020/4398061.

[7] Y. Lu, X. Ma, and C. Cui, "DCCS: A dual congestion control signals based TCP for datacenter networks," *Computer Networks*, vol. 247, pp. 110457–110457, Apr. 2024, doi: https://doi.org/10.1016/j.comnet.2024.110457.

[8]S. L. Yadav, R. L. Ujjwal, S. Kumar, O. Kaiwartya, M. Kumar, and P. K. Kashyap, "Traffic and Energy Aware Optimization for Congestion Control in Next Generation Wireless Sensor Networks," *Journal of Sensors*, vol. 2021, pp. 1–16, Jun. 2021, doi: https://doi.org/10.1155/2021/5575802.

[9]A. Ghaffari, "Congestion control mechanisms in wireless sensor networks: A survey," *Journal of Network and Computer Applications*, vol. 52, pp. 101–115, Jun. 2015, doi: https://doi.org/10.1016/j.jnca.2015.03.002.

[10]O. Gurewitz, M. Shifrin, and E. Dvir, "Data Gathering Techniques in WSN: A Cross-Layer View," *Sensors*, vol. 22, no. 7, p. 2650, Mar. 2022, doi: https://doi.org/10.3390/s22072650.

[11]Y. Lu, Z. Ling, S. Zhu, and L. Tang, "SDTCP: Towards Datacenter TCP Congestion Control with SDN for IoT Applications," *Sensors*, vol. 17, no. 12, p. 109, Jan. 2017, doi: https://doi.org/10.3390/s17010109.

[12]H. Liu, H. Ni, and R. Han, "A Transmission Rate Control Method for Active Congestion Reduction Based on Network Node Bandwidth Allocation," *Future Internet*, vol. 15, no. 12, pp. 385–385, Nov. 2023, doi: https://doi.org/10.3390/fi15120385.

[13]S. Ahmadi, "Security Implications of Edge Computing in Cloud Networks," *Journal of Computer and Communications*, vol. 12, no. 02, pp. 26–46, 2024, doi: https://doi.org/10.4236/jcc.2024.122003.

[14] Y. Shao *et al.*, "An energy-efficient distributed computation offloading algorithm for ground-air cooperative networks," *Vehicular Communications*, pp. 100875–100875, Jan. 2025, doi: https://doi.org/10.1016/j.vehcom.2025.100875.

[15] Q. Xu, G. Zhang, and J. Wang, "Research on Cloud-Edge-End Collaborative Computing Offloading Strategy in the Internet of Vehicles Based on the M-TSA Algorithm," *Sensors*, vol. 23, no. 10, pp. 4682–4682, May 2023, doi: https://doi.org/10.3390/s23104682.

[16] M. Abbasi, E. Mohammadi Pasand, and M. R. Khosravi, "Workload Allocation in IoT-Fog-Cloud Architecture Using a Multi-Objective Genetic Algorithm," *Journal of Grid Computing*, vol. 18, no. 1, pp. 43–56, Jan. 2020, doi: https://doi.org/10.1007/s10723-020-09507-1.

[17]O. H. Ahmed, J. Lu, Q. Xu, A. M. Ahmed, A. M. Rahmani, and M. Hosseinzadeh, "Using differential evolution and Moth–Flame optimization for scientific workflow scheduling in fog computing," *Applied Soft Computing*, vol. 112, p. 107744, Nov. 2021, doi: https://doi.org/10.1016/j.asoc.2021.107744.

[18] R. Ranji, A. Mansoor, and Asmiza Abdul Sani, "EEDOS: an energy-efficient and delay-aware offloading scheme based on device to device collaboration in mobile edge computing," *Telecommunication Systems*, vol. 73, no. 2, pp. 171–182, Feb. 2020, doi: https://doi.org/10.1007/s11235-019-00595-3.

[19] A. Asghari, Hossein Azgomi, Ali Abbas Zoraghchian, and Abbas Barzegarinezhad, "Energy-aware server placement in mobile edge computing using trees social relations optimization algorithm," *The Journal of Supercomputing*, Oct. 2023, doi: https://doi.org/10.1007/s11227-023-05692-4.

[20] A. Mondal, P. S. Chatterjee, and N. K. Ray, "An Optimal Novel Approach for Dynamic Energy-Efficient Task Offloading in Mobile Edge-Cloud Computing Networks," *SN Computer Science*, vol. 5, no. 5, Jun. 2024, doi: https://doi.org/10.1007/s42979-024-02992-1.

[21] X. Yang, "Optimizing Accounting Informatization through Simultaneous Multi-Tasking across Edge and Cloud Devices using Hybrid Machine Learning Models," *Journal of Grid Computing*, vol. 22, no. 1, Jan. 2024, doi: https://doi.org/10.1007/s10723-023-09735-1.

[22] M. I. Khaleel, M. Safran, S. Alfarhood, and D. Gupta, "Combinatorial metaheuristic methods to optimize the scheduling of scientific workflows in green DVFS-enabled edge-cloud computing," *Alexandria Engineering Journal*, vol. 86, pp. 458–470, Dec. 2023, doi: https://doi.org/10.1016/j.aej.2023.11.074.

[23] K. M. Hosny, A. I. Awad, W. Said, M. Elmezain, E. R. Mohamed, and M. M. Khashaba, "Enhanced whale optimization algorithm for dependent tasks offloading problem in multi-edge cloud computing," *Alexandria Engineering Journal*, vol. 97, pp. 302–318, Jun. 2024, doi: https://doi.org/10.1016/j.aej.2024.04.038.\

[24] Dr. S. R., S. J., V. P.N., and Dr. S. R., "Generalized Pascal's Triangle and its Properties," *NeuroQuantology*, vol. 20, no. 5, pp. 729–732, May 2022, doi: https://doi.org/10.14704/nq.2022.20.5.nq22229.

[25] J. B. Kadane, "Two Kadane Algorithms for the Maximum Sum Sub-Array Problem," Oct. 2023, doi: https://doi.org/10.20944/preprints202310.1061.v1.

[26] V. G. Kulkarni and N. Gautam, "Leaky buckets: sizing and admission control," *Proceedings of 35th IEEE Conference on Decision and Control*, doi: https://doi.org/10.1109/cdc.1996.574481.

[27] Dr. S. R., S. J., V. P.N., and Dr. S. R., "Generalized Pascal's Triangle and its Properties," *NeuroQuantology*, vol. 20, no. 5, pp. 729–732, May 2022, doi: https://doi.org/10.14704/nq.2022.20.5.nq22229.

[28] H. Fu, M. Sun, B. He, J. Li, and X. Zhu, "A Survey of Traffic Shaping Technology in Internet of Things," *IEEE access*, vol. 11, pp. 3794–3809, Jan. 2023, doi: https://doi.org/10.1109/access.2022.3233394.

[29] Miltiadis Alamaniotis and M. Alexiou, "Synergism of Fuzzy Leaky Bucket with Virtual Buffer for Large Scale Social Driven Energy Allocation in Emergencies in Smart City Zones," *Electronics*, vol. 13, no. 4, pp. 762–762, Feb. 2024, doi: https://doi.org/10.3390/electronics13040762.

[30] S. Laso, P. Rodríguez, J. L. Herrera, J. Berrocal, and J. M. Murillo, "Energy consumption and workload prediction for edge nodes in the Computing Continuum," *Sustainable Computing Informatics and Systems*, pp. 101088–101088, Jan. 2025, doi: https://doi.org/10.1016/j.suscom.2025.101088.

[31] H. H. Al-Mahmood and S. N. Alsaad, "Innovative Lightweight Encryption Schemes Leveraging Chaotic Systems for Secure Data Transmission," *Intelligent Automation & Soft Computing*, vol. 0, no. 0, pp. 1–10, Jan. 2024, doi: https://doi.org/10.32604/iasc.2024.059691.