

# Improved random early detection congestion control algorithm for internet routers

Samuel O. Hassan<sup>1</sup>, Adewole U. Rufai<sup>2</sup>, Michael O. Agbaje<sup>3</sup>, Theophilus A. Enem<sup>4</sup>,  
Lukman A. Ogundele<sup>1</sup>, Suleiman A. Usman<sup>4</sup>

<sup>1</sup>Department of Mathematical Sciences, Olabisi Onabanjo University, Ago-Iwoye, Nigeria

<sup>2</sup>Department of Computer Sciences, University of Lagos, Akoka, Lagos, Nigeria

<sup>3</sup>Department of Computer Science, Babcock University, Ilishan-Remo, Nigeria

<sup>4</sup>Department of Cyber Security, Air Force Institute of Technology, Kaduna, Nigeria

---

## Article Info

### Article history:

Received Dec 13, 2021

Revised Jul 9, 2022

Accepted Jul 27, 2022

---

### Keywords:

AQM algorithm  
Congestion control  
IM-RED algorithm  
Routers  
Simulation

---

## ABSTRACT

In the internet, router plays a strategic role in the transmission of data packets. Active queue management (AQM) aimed at managing congestion by keeping a reduced average buffer occupancy and hence a minimal delay. The novel random early detection (RED) algorithm suffers from large average buffer occupancy and delay shortcomings. This problem is due in part to the existence of a distinctive linear packet drop function it deploys. In this paper, we present a new version of RED, called improved RED (IM-RED). An important strategy of IM-RED is to deploy two dropping functions: i) nonlinear (i.e. quadratic) to deal with both light-and moderate-network traffic load conditions, and ii) linear to deal with heavy traffic load condition. Simulation experiments conducted using open-source ns-3 software to evaluate and compare the functionality of the proposed IM-RED with other two previous AQM algorithms confirmed that IM-RED reduces the average buffer occupancy and obtained an improved delay performance especially at heavy network traffic load scenario. Very fortunately, since RED algorithm is known to appear as a built-in model in ns-3 and even Linux kernel, its implementation can therefore be leveraged to obtain IM-RED while only adjusting the packet dropping probability profile and holding on to its other attributes.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



---

### Corresponding Author:

Samuel O. Hassan  
Department of Mathematical Sciences, Olabisi Onabanjo University  
Ago-Iwoye, Nigeria  
Email: samuel.hassan@oouagoiwoye.edu.ng

---

## 1. INTRODUCTION

The massive development of digital technologies and continuous rise in the number of Internet users leads to the problem of inevitable growth of congestion in the current internet [1]–[4]. Network congestion is a situation in which the size/amount of generated data traffic exceeds the capacity of network's resource. In the Internet, a key resource like routers play a crucial role of handling large traffic loads is affected by congestion [3]. Congestion has an adverse effect of degrading network performance [5]–[7]. Therefore, avoidance/control of congestion is imperative in order to ensure an improved quality of service for users on the Internet [7], [8]. Traditionally in Internet routers, DropTail algorithm drops packet only when the buffer is full. Some of the drawback of this approach includes large delay, buffer overflow problems, global synchronization, and network deadlock [9]. By contrast, active queue management (AQM) algorithms detects

incipient congestion early and proactively sends a congestion feedback (through a packet dropped) to end-systems to regulate their transmission rate before overflow occurs in the queue of the buffer [10]–[12].

The random early detection (RED) algorithm developed by Floyd and Jacobson [13] in the early 90’s is a typical form of AQM algorithm. RED, which yields an improved results than DropTail has been recommended by internet engineering task force (IETF) for implementation in Internet routers for congestion control [9]. RED has the following associated important four parameters:  $THmin$  (that is, lower queue threshold position),  $THmax$  (that is, upper queue threshold position),  $max_p$  (that is, the maximum packet drop probability), and  $w_q$  (that is, the weight factor:  $0 < w_q < 1$ ).

One important aspect of RED is the computation of average queue size ( $qAvg$ ) (that is, the average buffer occupancy) which is used as an indicator for congestion. A low-pass filter exponential weighted moving average (EWMA) is usually employed for the computation of  $qAvg$  according to (1) for each packet arrival into the router:

$$qAvg = \begin{cases} (1 - w_q) \times qAvg' + (w_q \times q), & q \neq 0 \\ (1 - w_q)^m \times qAvg', & \text{otherwise} \end{cases} \tag{1}$$

where:

$q$  indicates the current queue size,  $qAvg'$  indicates the computed old average buffer occupancy,  $w_q$  ( $0 < w_q < 1$ ) is a pre-defined weight parameter to calculate average buffer occupancy, and  $m$  which refers to the idle time parameter is computed according to (2):

$$m = f(\text{time} - q\_idle\_time) \tag{2}$$

where  $q\_idle\_time$  indicates the beginning of the queue idle time.

Subsequently, a decision is made whether to enqueue or reject the packet. In a situation where the numerical value of the average buffer occupancy is lesser than  $THmin$ , all incoming packets will be accepted and inserted into the queue. In cases when  $THmin \leq qAvg < THmax$ , arriving packets are linearly dropped; this linear function rises from 0 to  $max_p$ . However, in cases when the average buffer occupancy is greater than  $THmax$ , all arriving packets are dropped with probability 1. The packet drop function for RED is given as:

$$P_b = \begin{cases} 0 & \text{for } qAvg < THmin \\ max_p \left( \frac{qAvg - THmin}{THmax - THmin} \right) & \text{for } THmin \leq qAvg < THmax \\ 1 & \text{for } THmax \leq qAvg \end{cases} \tag{3}$$

such that,

$$P_a = P_b \left( \frac{1}{1 - count \times P_b} \right) \tag{4}$$

where  $P_b$  indicates the initial packet dropping probability;  $P_a$  indicates the final packet dropping probability and  $count$  denote the number of arrived packets since the last dropped.

One major weakness of RED, specifically as it relates to the distinctive linear drop function utilized when the average buffer occupancy varies between  $THmin$  and  $THmax$  is that it tends to behave too aggressively when traffic load is light (that is, when network congestion is not serious) and not aggressive enough at heavy traffic load (when congestion in the network is serious) [14]. In fact, Paul *et al.* [15] described the distinctive linear drop function of RED as inadequate. Hitherto, RED being the oldest form of AQM algorithm remains a vastly studied and improved algorithm [6], [14], [16]–[18]. Accordingly, its common to find an enormous RED-oriented algorithms in literature.

The nonlinear RED (NLRED) algorithm in Zhou *et al.* [14] has turned out to perform better than RED in terms of throughput due to the presence of a quadratic drop function utilized when  $THmin \leq qAvg < THmax$ . This quadratic function rises from 0 to  $max'_p$  (in which  $max'_p = 1.5 \times max_p$ ). The packet drop function for NLRED is expressed according to (5).

$$P_b = \begin{cases} 0 & \text{for } qAvg < THmin \\ max'_p \left( \frac{avg - THmin}{THmax - THmin} \right)^2 & \text{for } THmin \leq qAvg < THmax \\ 1 & \text{for } THmax \leq qAvg \end{cases} \tag{5}$$

The gentle RED (GRED) suggested in Floyd [19] can be rightly described as an improved version of RED algorithm such that when the average buffer occupancy exceeds  $TH_{max}$ , packets are not forced drop. In GRED, when  $TH_{min} \leq qAvg < 2 \times TH_{max}$ , arriving packets are linearly dropped. This linear function rises from  $max_p$  to 1. The packet drop function for GRED is expressed according to (6).

$$P_b = \begin{cases} 0 & \text{for } qAvg < TH_{min} \\ max_p \left( \frac{qAvg - TH_{min}}{TH_{max} - TH_{min}} \right) & \text{for } TH_{min} \leq qAvg < TH_{max} \\ max_p + (1 - max_p) \left( \frac{qAvg - TH_{min}}{TH_{max}} \right) & \text{for } TH_{max} \leq qAvg < 2 \times TH_{max} \\ 1 & \text{for } 2 \times TH_{max} \leq qAvg \end{cases} \quad (6)$$

The strategy of authors in Zhang *et al.* [20] in contrast to GRED is to quadratically drop packets randomly when  $TH_{min} \leq qAvg < TH_{max}$ . This quadratic function increases from 0 to  $max_p$ . The packet drop function for MRED is expressed according to (7).

$$P_b = \begin{cases} 0 & \text{for } qAvg < TH_{min} \\ max_p \left( \frac{qAvg^2 - TH_{min}^2}{TH_{max}^2 - TH_{min}^2} \right) & \text{for } TH_{min} \leq qAvg < TH_{max} \\ max_p + (1 - max_p) \left( \frac{qAvg - TH_{min}}{TH_{max}} \right) & \text{for } TH_{max} \leq qAvg < 2 \times TH_{max} \\ 1 & \text{for } 2 \times TH_{max} \leq qAvg \end{cases} \quad (7)$$

To tackle the shortcomings of high number of dropped packets when the average buffer occupancy exceeds  $TH_{max}$  in RED, Prabhavat and Varakulsiripunth [21] proposed extended drop slope RED (ExRED) which utilizes a 2<sup>nd</sup> order polynomial drop function. The packet drop function for ExRED is expressed according to (8).

$$P_b = \begin{cases} 0 & \text{for } qAvg < TH_{min} \\ max_p \left( \frac{qAvg - TH_{min}}{TH_{max} - TH_{min}} \right) & \text{for } TH_{min} \leq qAvg < TH_{max} \\ a_2 qAvg^2 + a_1 qAvg + a_0 & \text{for } TH_{max} \leq qAvg < K \end{cases} \quad (8)$$

where:

$K$  refers to the buffer size,

$$a_2 = \frac{(TH_{max} - TH_{min}) - (K - TH_{min})max_p}{(TH_{max} - TH_{min})(K - TH_{max})^2} \quad (9)$$

$$a_1 = \frac{(TH_{max}^2 + K^2 - 2TH_{max}TH_{min})max_p - 2TH_{max}^2 + 2TH_{max}TH_{min}}{(TH_{max} - TH_{min})(K - TH_{max})^2} \quad (10)$$

$$a_0 = \frac{TH_{max}^3 - TH_{max}^2 TH_{min} - (TH_{max}^2 + 2TH_{max}TH_{min} + TH_{min}K)Kmax_p}{(TH_{max} - TH_{min})(K - TH_{max})^2} \quad (11)$$

Another variation of RED named double slope RED (DSRED) in Zheng and Atiquzzaman [22], works in such a way that a two linear drop functions (with slopes  $\alpha$  and  $\beta$  expressed according to (12) and (13) respectively) was deployed.  $\alpha$  and  $\beta$  are complementary and adjustable through a mode selector ( $\gamma$ ). This was done in order to remedy the low throughput drawback of RED.

$$\alpha = \frac{2(1-\gamma)}{TH_{max} - TH_{min}} \quad (12)$$

$$\beta = \frac{2\gamma}{TH_{max} - TH_{min}} \quad (13)$$

In effect, the packet drop function for DSRED is expressed according to (14):

$$P_b = \begin{cases} 0 & \text{for } qAvg < TH_{min} \\ \alpha (qAvg - TH_{min}) & \text{for } TH_{min} \leq qAvg < TH_{med} \\ 1 - \gamma + \beta (qAvg - TH_{med}) & \text{for } TH_{med} \leq qAvg < TH_{max} \\ 1 & \text{for } TH_{max} \leq qAvg < BS \end{cases} \quad (14)$$

where  $TH_{med}$  is a mid-point threshold between  $TH_{min}$  and  $TH_{max}$  set as  $((TH_{min} + TH_{max})/2)$ ;  $BS$  indicates the buffer size.

Hassan and Oluwatope [23] believed that one approach for alleviating the large delay weakness of RED was to modify its dropping function. To achieve this, curvilinear RED (CLRED) algorithm was proposed which can be defined according to (15):

$$P_b = \begin{cases} 0, qAvg < TH_{min} \\ 4(1 - max_p) \left( \frac{qAvg - TH_{min}}{TH_{max} - TH_{min}} \right)^2, TH_{min} \leq qAvg < TH_{med} \\ (1 - max_p) + 2max_p \left( \frac{qAvg - TH_{med}}{TH_{max} - TH_{min}} \right), TH_{med} \leq qAvg < TH_{max} \\ 1, TH_{max} \leq qAvg \end{cases} \quad (15)$$

where  $TH_{med} = (TH_{min} + TH_{max})/2$  refers to the mid-point threshold between  $TH_{min}$  and  $TH_{max}$ ;

Congestion control RED (CoCo-RED) algorithm by Suwannapong and Khunboa [24] can also be described as an improvement-yielding algorithm over RED. Using CoCo-RED, when the average buffer occupancy is between  $TH_{min}$  and  $TH_{max}$ , packets are linearly dropped. This linear function rises from 0 to  $max_p$ . However, when the average buffer occupancy is between  $TH_{max}$  and the buffer capacity, packets are exponentially dropped. This exponential function rises from  $max_p$  to 1. The authors claimed that the exponential function will reduce the number of dropped packets especially when  $TH_{max} \leq qAvg$  (as it is the case with RED).

In an attempt to remedy the shortcomings of not too high aggressiveness of RED especially when the average buffer occupancy is near  $TH_{max}$  and to eliminate the reliance on  $max_p$  parameter, Abdel-Jaber [25] proposed RED\_E (RED-Exponential). The packet drop function for RED\_E is expressed according to (16):

$$P_b = \begin{cases} 0 \text{ for } qAvg < TH_{min} \\ \left( \frac{e^{qAvg} - e^{TH_{min}}}{e^{TH_{max}} - e^{TH_{min}}} \right) \text{ for } TH_{min} \leq qAvg < TH_{max} \\ 1 \text{ for } TH_{max} \leq qAvg \end{cases} \quad (16)$$

The underlying idea of smart RED (SmRED) in Paul *et al.* [15] is to improve RED by utilizing both quadratic and linear packet drop functions. SmRED was shown to achieved a trade-off between two important metrics: throughput and delay. The packet drop function for SmRED is expressed according to (17):

$$P_b = \begin{cases} 0 \text{ for } qAvg < TH_{min} \\ max_p \left( \frac{qAvg - TH_{min}}{TH_{max} - TH_{min}} \right)^2 \text{ for } TH_{min} \leq qAvg < Target \\ max_p \sqrt{\frac{qAvg - TH_{min}}{TH_{max} - TH_{min}}} \text{ for } Target \leq qAvg < TH_{max} \\ 1 \text{ for } TH_{max} \leq qAvg \end{cases} \quad (17)$$

where:

$$Target = TH_{min} + \left( \frac{TH_{max} - TH_{min}}{2} \right) \quad (18)$$

Paul *et al.* [26] proposed the SmRED- $i$  algorithm which modified SmRED by using different values for  $i$  parameter in an attempt to tune the packet drop function. The drop function for SmRED- $i$  is expressed according to (19):

$$P_b = \begin{cases} 0 \text{ for } qAvg < TH_{min} \\ max_p \left( \frac{qAvg - TH_{min}}{TH_{max} - TH_{min}} \right)^i \text{ for } TH_{min} \leq qAvg < Target \\ max_p \left( \frac{qAvg - TH_{min}}{TH_{max} - TH_{min}} \right)^{1/i} \text{ for } Target \leq qAvg < TH_{max} \\ 1 \text{ for } TH_{max} \leq qAvg \end{cases} \quad (19)$$

where  $i = 2, 3, 4, 5, \dots$

Patel and Karmeshu [27], the authors developed an improved variant of RED that exploit *count* variable in computing the drop probability function which is expressed as:

$$P_b = \begin{cases} 0 & \text{for } qAvg < THmin \\ 1 - \frac{p_1 \times (-\log(p_1))}{(count+1)} & \text{for } THmin \leq qAvg < THmax \\ 1 & \text{for } THmax \leq qAvg \end{cases} \quad (20)$$

in which:

$$p_1 = \max_p \left( \frac{avg-THmin}{THmax-THmin} \right) \quad (21)$$

The modified algorithm obtained an improved network performance by ensuring a reduced delay and high throughput when network load is heavy.

It is more reasonable to ascertain the effectiveness of a congestion control algorithm when the network traffic load is heavy [28]. In this paper, we propose a new flavor of RED, named improved random early detection (IM-RED) algorithm which utilizes two packet dropping functions in lieu of one drop function used in RED. This way, at light-and moderate-network traffic load conditions, the proposed IM-RED algorithm drops packet using a nonlinear (quadratic) drop function in order to act less aggressively. At heavy traffic load, IM-RED utilizes a linear drop function in order to act more aggressively.

The arrangement of the paper is as shown in; section 2 provides a description of the proposed IM-RED algorithm. Section 3 presents the simulation results. Finally, section 4 concludes the paper.

## 2. IMPROVED RED (IM-RED) AQM ALGORITHM

In the proposed improved random early detection (referred to as “IM-RED”) algorithm, the underlying strategy is to partition the queue domain between *THmin* and *THmax* (of RED algorithm which uses a linear function) into two (non-uniform) subintervals of uneven widths so as to distinguish light-and moderate-network traffic load conditions from a heavy network traffic load condition. In order to do this, a new *Target* value is added and expressed in the form (2).

$$Target = THmin + \left( \frac{THmax+THmin}{3} \right) \quad (22)$$

The packet drop function ( $P_b$ ) for IM-RED is expressed in the form:

$$P_b = \begin{cases} 0 & \text{for } qAvg < THmin \\ 9\max_p \left( \frac{qAvg-THmin}{THmax+THmin} \right)^2 & \text{for } THmin \leq qAvg < Target \\ \max_p + 3(1 - \max_p) \left( \frac{qAvg-Target}{2(THmax-2THmin)} \right) & \text{for } Target \leq qAvg < THmax \\ 1 & \text{for } THmax \leq qAvg \end{cases} \quad (23)$$

Figure 1 illustrates the packet dropping probability function curve for IM-RED. The proposed AQM algorithm computes the average buffer occupancy in accordance with (1) similar to RED and behave as follows:

- When the numerical value of the average buffer occupancy is less than *THmin*, then no incoming packet will be dropped. That is:

$$P_b = 0 \quad (24)$$

- The long region of the curve is utilized for light traffic load (when the average buffer occupancy is near *THmin*) and moderate traffic load (when the average buffer occupancy is close to *Target*). For this segment, IM-RED drops packet randomly using a quadratic drop function that changes from 0 to  $\max_p$  according to (25). At this state, congestion in the network is considered not too serious.

$$P_b = 9\max_p \left( \frac{qAvg-THmin}{THmax+THmin} \right)^2 \quad (25)$$

- The short region of the curve is utilized for heavy traffic load (when the average buffer occupancy is between *Target* and *THmax*). For this segment, IM-RED randomly drops packet more aggressively

using a linear packet drop function that changes from  $max_P$  to 1 according to (26). At this stage, network congestion is believed to be very serious. So, an aggressive drop function is needed in order to ease congestion.

$$P_b = max_P + 3(1 - max_P) \left( \frac{qAvg - Target}{2(THmax - 2THmin)} \right) \quad (26)$$

- d. When the numerical value of the average buffer occupancy exceeds  $THmax$ , then all incoming packet will be dropped with probability 1. That is (27).

$$P_b = 1 \quad (27)$$

We provide the pseudo-code for IM-RED in Algorithm 1.

#### Algorithm 1. Pseudo-code for IM-RED algorithm

```

1: Initialization:
2:   Set  $qAvg = 0$ 
3:   Set  $count = -1$ 
4:   For each packet arrival at IM-RED router buffer do
4: Calculate the average buffer occupancy,  $qAvg$ 
5: if the buffer of the router is nonempty then
6:  $qAvg = (1 - w_q) \times qAvg' + (w_q \times q)$ 
7: else if  $m = f(q\_current\_time - q\_idle\_time)$  then
8:  $qAvg = (1 - w_q)^m \times qAvg'$ 
9: end if
10: if  $qAvg < THmin$  then
11: No dropping of packets
12: Set  $count = count - 1$ 
13: else if  $THmin \leq qAvg < Target$  then
14: Set  $count = count + 1$ 
15: Calculate the packet drop probability  $P_a$ 
16:  $P_b = 9max_P \left( (qAvg - THmin) / (THmax + THmin) \right)^2$ 
17:  $P_a = P_b / (1 - count \times P_b)$ 
18: Mark/drop the arriving packet with probability  $P_a$ 
19: Set  $count = 0$ 
20: Drop the packet
21: else if  $Target \leq qAvg < THmax$  then
22: Set  $count = count + 1$ 
23: Calculate the packet drop probability  $P_a$ 
24:  $P_b = max_P + 3(1 - max_P) \left( (qAvg - Target) / 2(THmax - 2THmin) \right)$ 
25:  $P_a = P_b / (1 - count \times P_b)$ 
26: Mark/drop the arriving packet with probability  $P_a$ 
27: Set  $count = 0$ 
28: Drop the packet
29: else if  $THmax \leq qAvg$  then
30: Drop the arriving packet
31: Set  $count = 0$ 
32: end if
33: if  $count = -1$  then
34: When the buffer of the router becomes empty
35: Set  $q\_idle\_time = time$ 
36: end if
37:
38: Saved Variables:
39:  $qAvg$ : average buffer occupancy
40:  $q\_idle\_time$ : beginning of queue idle time
41:  $count$ : number of arrived packets since the last dropped
42:
43: Preset input parameters:
44:  $THmin$ : lower queue threshold position
45:  $THmax$ : upper queue threshold position
46:  $max_P$ : maximum packet drop probability
47:  $w_q$ : pre-defined weight parameter
48:
49: Other:
50:  $P_a$ : current packet marking probability
51:  $q$ : current buffer occupancy
52:  $q\_current\_time$ : current time
53:  $f(t)$ : a linear function of time  $t$ 

```

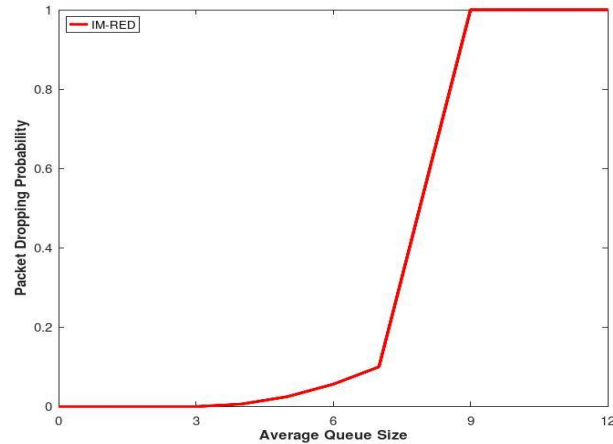


Figure 1. IM-RED's packet dropping probability

### 3. SIMULATION RESULTS AND DISCUSSION

This section evaluates the functionality of IM-RED AQM algorithm under light, moderate, and heavy network traffic loads and compare it with two algorithms: NLRED and RED in ns-3 simulator [29]. The network topology simulated is shown in Figure 2. The topology consists of  $N$  sources that transmits to a sink via an intermediate node (i.e. the router). Each of the sources is connected to the router using a link rate of 100 Mbps with 4 ms delay time. The router (having the implementation for the AQM algorithms) on the other hand is connected to the sink using a bottleneck link rate of 10 Mbps (which corresponds to 1250 packet/second with the average packet size of 1000 bytes) with 10 ms delay time. The buffer size is 25 packets. By varying the number of sources, different levels of traffic load and thus varying levels of congestion are produced on the bottleneck link. Other configuration settings are presented in Table 1.

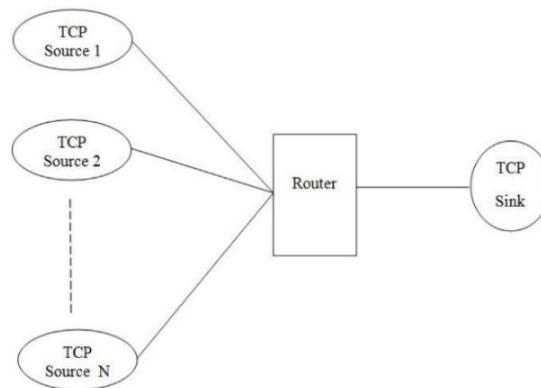


Figure 2. Network topology

Table 1. Configuration settings

Simulation parameter	Value
$TH_{min}$	3 packets
$Target$	7 packets
$TH_{max}$	9 packets
$max_p$	0.1
$w_q$	0.002
TCP	NewReno
Simulation time	100 s

#### 3.1. Light traffic load

In this subsection, the scenario considered comprises of a dumbbell topology with 5 transmission control protocol (TCP) flows transmitting across a single/shared bottleneck link to a sink. This experiment is meant to assess and establish how all the three algorithms deals with light traffic congestion. Figure 3 shows

the comparative analysis of the algorithms. Figure 3(a) depicts the variations in instantaneous average queue size over time for RED, NLRED, and IM-RED algorithms. As can be noticed from the figure, the initial climax of RED gets as far as 9.5372, NLRED gets as far as 9.5372, while IM-RED gets as far as 7.0293. Analysis presented in Table 2 shows that IM-RED yields improved performance (with lower average queue size) than the two other algorithms. This is because, at light load when there is less congestion in the network, IM-RED minimizes the packet dropping probability. Figure 3(b) depicts the instantaneous delay for NLRED, RED and IM-RED algorithms. Analysis presented in Table 3 shows that NLRED obtained the best performance, although IM-RED performed better than RED. Figure 3(c) shows the throughput for NLRED, RED and IM-RED algorithms. As presented in the analysis of Table 4, all the three algorithms performs similar in terms of throughput.

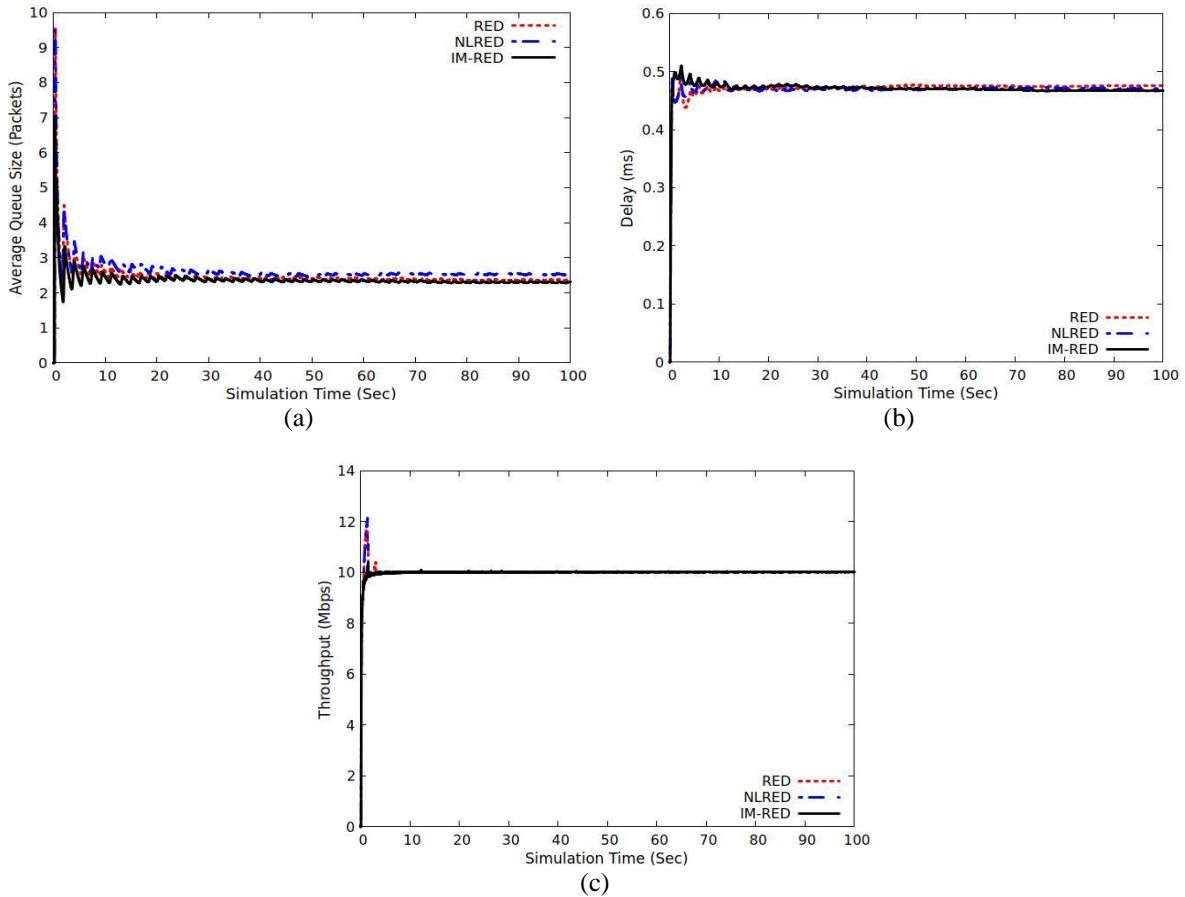


Figure 3. Light traffic condition: (a) average queue size graph, (b) delay graph, and (c) throughput graph

Table 2. Comparison of average queue size (packets)

AQM Algorithm	Light Traffic Mean	Moderate Traffic Mean	Heavy Traffic Mean
RED	2.4736	5.3286	5.9357
NLRED	2.6179	5.1199	5.5521
IM-RED	2.3668	4.4252	4.8948

Table 3. Comparison of delay (ms)

AQM Algorithm	Light Traffic Mean	Moderate Traffic Mean	Heavy Traffic Mean
RED	0.4712	2.1149	5.2364
NLRED	0.4688	2.0766	5.1146
IM-RED	0.4701	2.0119	5.1045



Table 4. Comparison of throughput (Mbps)

AQM algorithm	Light traffic mean	Moderate traffic mean	Heavy traffic mean
RED	9.9975	10.0188	10.1668
NLRED	9.9971	10.0259	10.1477
IM-RED	9.9851	10.0201	10.0965

### 3.2. Moderate traffic load

In this subsection, the scenario considered comprises of a dumbbell topology with 20 TCP flows transmitting across a single/shared bottleneck link to a sink. This experiment is meant to assess and establish how all the three algorithms deals with moderate traffic congestion. Figure 4 shows the comparative analysis of the algorithms. Figure 4(a) depicts the variations in instantaneous average queue size over time of NLRED, RED and IM-RED algorithms. As can be noticed from the figure, the initial climax of all the three algorithms gets as far as 6.6667. Results presented in Table 2 shows that IM-RED offers an improved performance (with smaller average queue size) than the two other algorithms. This is because when the network is moderately congested, IM-RED utilizes a smaller packet dropping probability. Figure 4(b) shows the instantaneous delay for NLRED, RED and IM-RED algorithms. Further results presented in Table 3 shows that again IM-RED obtained the best performance. Figure 4(c) shows the throughput for NLRED, RED and IM-RED algorithms. As presented in the statistics of Table 4, all the three algorithms (optimally) performs similar in terms of throughput.

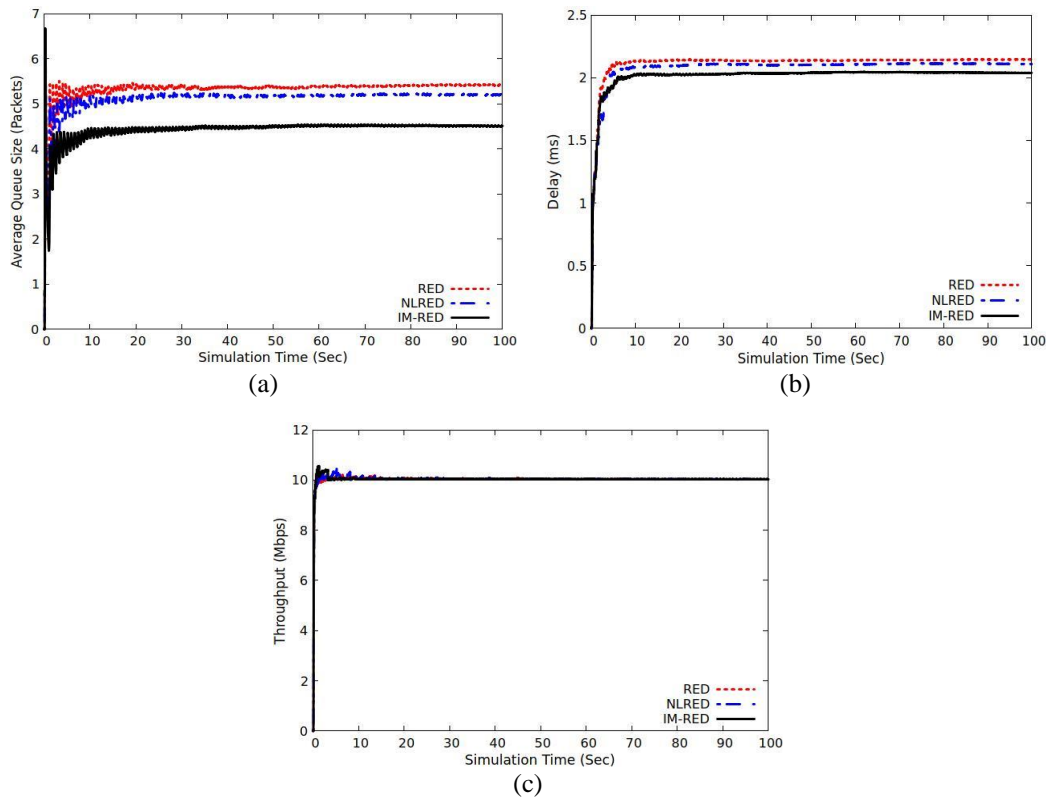


Figure 4. Moderate traffic condition: (a) average queue size graph, (b) delay graph, and (c) throughput graph

### 3.3. Heavy traffic load

In this subsection, the scenario considered comprises of a dumbbell topology with 50 TCP flows transmitting across a single/shared bottleneck link to a sink. This experiment is meant to assess and establish how all the three algorithms deals with heavy traffic congestion. Figure 5 shows the comparative analysis of the algorithms. Figure 5(a) depicts the variations in instantaneous average queue size over time of RED, NLRED and IM-RED algorithms. As can be observed from the figure, the initial climax of RED gets as far as 6.1904, NLRED gets as far as 5.7693 while IM-RED gets as far as 5.1614. As can be noticed in the analysis presented in Table 2, that IM-RED achieved a better performance (with lower average queue size) than the two other algorithms. This is because when the network congestion is serious, IM-RED utilized a higher

packet dropping probability than others. Figure 5(b) shows the instantaneous delay for NLRED, RED and IM-RED algorithms. Analysis presented in Table 3 shows that a IM-RED outperformed all two others by obtaining the lowest average value for delay. This confirms the aggressive packet dropping property of IM-RED at heavy traffic load in order to ease congestion. Figure 5(c) shows the throughput for NLRED, RED and IM-RED algorithms. Analysis presented in Table 4 shows that all the three algorithms performs similar in terms of throughput.

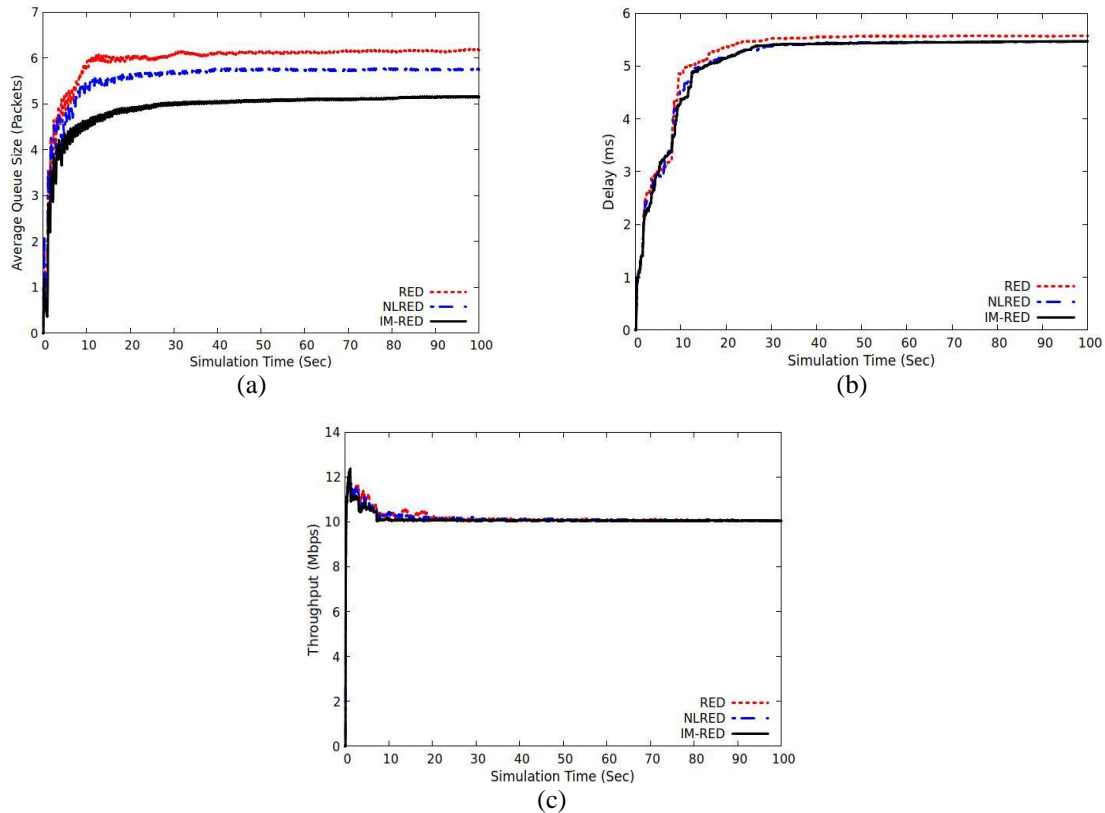


Figure 5. Heavy traffic condition: (a) average queue size graph, (b) delay graph, and (c) throughput graph

#### 4. CONCLUSION

This paper has presented an improved RED algorithm named IM-RED. This algorithm emerged from inspiration drawn following the simple design of RED model and deploys a two packet dropping functions. Specifically, a nonlinear quadratic packet drop function is utilized (to ensure a less aggressive packet drop action) for a light and moderate traffic load conditions while a linear drop function is deployed (to ensure a more aggressive packet drop action) for a heavy traffic load condition. Results from simulation experiments revealed that IM-RED stabilizes and maintains a small average buffer occupancy. At heavy traffic load, IM-RED obtained an improved delay performance. In our near future work, we intend to implement IM-RED for congestion control in Constrained Application Protocol (CoAP) Observe Group Communication.




#### REFERENCES

- [1] L. Pei, F. Wu, and S. Wang, "Periodic, quasi-periodic and chaotic oscillations in two heterogeneous AIMD/RED Network congestion models with state-dependent round-trip delays," *International Journal of Bifurcation and Chaos*, vol. 31, no. 6, p. 2150124, May 2021, doi: 10.1142/S0218127421501248.
- [2] Y. Su, L. Huang, and C. Feng, "QRED: A Q-learning-based active queue management scheme," *Journal of Internet Technology*, vol. 19, no. 4, pp. 1169–1178, 2018, doi: 10.3966/160792642018081904019.
- [3] A. Adamu, Y. Surajo, and M. T. Jafar, "Sared: Self-adaptive active queue management scheme for improving quality of service in network systems," *Computer Science*, vol. 22, no. 2, pp. 253–267, Apr. 2021, doi: 10.7494/csci.2021.22.2.4020.
- [4] G. Attiya and H. El-Khobby, "Improving internet quality of service through active queue management in routers," *International Journal of Computer Science*, vol. 9, no. 1, pp. 279–286, 2012, [Online]. Available: <http://www.ijcsi.org/papers/IJCSI-9-1-2-279-286.pdf>.
- [5] M. Baklizi, "Weight queue dynamic active queue management algorithm," *Symmetry*, vol. 12, no. 12, pp. 1–16, Dec. 2020, doi: 10.3390/sym12122077.





- [6] A. A. Abu-Shareha, "Controlling delay at the router buffer using modified random early detection," *International Journal of Computer Networks and Communications*, vol. 11, no. 6, pp. 63–75, Nov. 2019, doi: 10.5121/IJCNC.2019.11604.
- [7] A. Adamu, V. Shorgin, S. Melnikov, and Y. Gaidamaka, "Flexible Random early detection algorithm for queue management in routers," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12563 LNCS, 2020, pp. 196–208.
- [8] M. M. Hamdi, H. F. Mahdi, M. S. Abood, R. Q. Mohammed, A. D. Abbas, and A. H. Mohammed, "A review on Queue Management Algorithms in Large Networks," *IOP Conference Series: Materials Science and Engineering*, vol. 1076, no. 1, p. 012034, 2021, doi: 10.1088/1757-899x/1076/1/012034.
- [9] B. Braden *et al.*, "Recommendations on queue management and congestion avoidance in the internet," Apr. 2016. doi: 10.17487/rfc2309.
- [10] S. Kar, B. Alt, H. Koepl, and A. Rizk, "PAQMAN: A Principled approach to active queue management," *arxiv*, Feb. 2022, [Online]. Available: <http://arxiv.org/abs/2202.10352>.
- [11] M. Barczyk and A. Chydzinski, "AQM based on the queue length: A real network study," *PLoS ONE*, vol. 17, no. 2 February, p. e0263407, Feb. 2022, doi: 10.1371/journal.pone.0263407.
- [12] J. Aweya, M. Ouellette, and D. Y. Montuno, "A control theoretic approach to active queue management," *Computer Networks*, vol. 36, no. 2–3, pp. 203–235, Jul. 2001, doi: 10.1016/S1389-1286(00)00206-1.
- [13] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993, doi: 10.1109/90.251892.
- [14] K. Zhou, K. L. Yeung, and V. O. K. Li, "Nonlinear RED: A simple yet efficient active queue management scheme," *Computer Networks*, vol. 50, no. 18, pp. 3784–3794, Dec. 2006, doi: 10.1016/j.comnet.2006.04.007.
- [15] A. K. Paul, H. Kawakami, A. Tachibana, and T. Hasegawa, "An AQM based congestion control for eNB RLC in 4G/LTE network," in *Canadian Conference on Electrical and Computer Engineering*, May 2016, vol. 2016–October, pp. 1–5, doi: 10.1109/CCECE.2016.7726792.
- [16] Z. Albayrak and M. Çakmak, "A review: active queue management algorithms in mobile communication," *International Conference on Cyber Security and Computer Science*, no. October 2018, pp. 180–184, 2018.
- [17] N. Hamadneh, M. Obiedat, A. Qawasmeh, and M. Bsoul, "HRED, an active queue management algorithm for TCP congestion control," *Recent Patents on Computer Science*, vol. 12, no. 3, pp. 212–217, May 2018, doi: 10.2174/2213275912666181205155828.
- [18] J. M. Amigó, A. Giménez, O. M. Bonastre, and J. Valero, "Internet congestion control: from stochastic to dynamical models," *Stochastics and Dynamics*, vol. 21, no. 3, Jan. 2021, doi: 10.1142/S0219493721400098.
- [19] S. Floyd, "Recommendation on using the gentle variant of RED," 2000. [Online]. Available: [https://www.icir.org/floyd/red/gentle.html#:~:text=For the best behavior of,gentle\\_%22 parameter set to 1.](https://www.icir.org/floyd/red/gentle.html#:~:text=For the best behavior of,gentle_%22 parameter set to 1.)
- [20] Y. Zhang, J. Ma, Y. Wang, and C. Xu, "MRED: an improved nonlinear RED Algorithm," in *International Conference Proceedings on Computer and Automation Engineering (ICCAE 2011)*, 2012, vol. 44, no. Iccae 2011, pp. 6–11.
- [21] S. Prabhavat and R. Varakulsiripunth, "Performance improvement on RED based gateway in TCP communication network," in *Proceedings of the International Conference on Control, Automation Systems (ICCAS 2004)*, 2004, pp. 782–787.
- [22] B. Zheng and M. Atiquzzaman, "DSRED: an active queue management scheme for next generation networks," in *Proceedings 25th Annual IEEE Conference on Local Computer Networks. LCN 2000*, 2000, pp. 242–251, doi: 10.1109/LCN.2000.891036.
- [23] S. Hassan and A. O. Oluwatope, "Curvilinear RED: an improved red algorithm for internet routers," in *Environment and Water Resource Management / 813: Modelling and Simulation / 814: Power and Energy Systems / 815: Health Informatics*, 2014, pp. 154–160, doi: 10.2316/P.2014.813-025.
- [24] Suwannapong and Khunboa, "Congestion control in CoAP observe group communication," *Sensors*, vol. 19, no. 15, p. 3433, Aug. 2019, doi: 10.3390/s19153433.
- [25] H. Abdel-Jaber, "An Exponential active queue management method based on random early detection," *Journal of Computer Networks and Communications*, vol. 2020, pp. 1–11, May 2020, doi: 10.1155/2020/8090468.
- [26] A. Paul, H. Kawakami, A. Tachibana, and T. Hasegawa, "Effect of AQM-based RLC buffer management on the eNB scheduling algorithm in LTE Network," *Technologies*, vol. 5, no. 3, p. 59, Sep. 2017, doi: 10.3390/technologies5030059.
- [27] S. Patel and Karmeshu, "A New modified dropping function for congested AQM networks," *Wireless Personal Communications*, vol. 104, no. 1, pp. 37–55, Jan. 2019, doi: 10.1007/s11277-018-6007-8.
- [28] P. K. Dash, N. K. Barpanda, and M. Panda, "Congestion control in cable network transmission using novel RED algorithm," *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 10, pp. 2322–2324, Aug. 2019, doi: 10.35940/ijitee.I9562.0881019.
- [29] University of Washington NS-3 Consortium, "ns-3 Network Simulator," NS-3, 2022. [Online]. Available: <https://www.nsnam.org/> (accessed Oct. 05, 2021).

## BIOGRAPHIES OF AUTHORS







**Samuel O. Hassan**    received his M.Sc. and Ph.D degrees in Computer Science from Obafemi Awolowo University, Ile-Ife, Nigeria. Currently, he is a Lecturer in the Department of Mathematical Sciences (Computer Science Unit), Olabisi Onabanjo University, Ago-Iwoye, Nigeria. He is a Certified Information Technology Practitioner (C.itp). He is a Member of Computer Professionals (Registration Council) of Nigeria (CPN), International Association of Engineers (IAENG). He is also a member of the following societies: Nigeria Computer Society (NCS), the IAENG Society of Computer Science, the IAENG Society of Internet Computing and Web Services, the IAENG Society of Scientific Computing, the IAENG Society of Software Engineering, and the IAENG Society of Wireless Networks. He is also a Member of MathTech Thinking Foundation, (MTTF) and a certified MTTF-Advisor. His research interests spans Computational Mathematics, computer networks and communications, Mathematical modeling and simulation, and Internet congestion control. He can be contacted at email: [samuel.hassan@oouagoiwoye.edu.ng](mailto:samuel.hassan@oouagoiwoye.edu.ng).







**Adewole U. Rufai**     is a Senior Lecturer in the Department of Computer Sciences, University of Lagos, Nigeria. His tertiary education includes the following; Bachelor of Science degree in Mathematics obtained from the University of Ilorin, PGD, M.Sc., Ph.D. (Computer Science), and MBA (Finance) all from the University of Lagos. He started his career in academia at Olabisi Onabanjo University, Ago-Iwoye as a Graduate Assistant (2000-2002), Assistant Lecturer (2002-2005), and Lecturer II (2005-2006). He later joined the University of Lagos in 2006 and rose through the ranks to become a Senior Lecturer. Dr. Rufai is a Fellow of the Nigeria Computer Society, a Certified Information Technology Practitioner, and a Member of the Computer Professionals Registration Council of Nigeria and a Member of Association of Computer Machinery. His areas of research include: Software Engineering, Cognitive Computing, and Cyber Security. He can be contacted at email: arufai@unilag.edu.ng.







**Michael O. Agbaje**     is an Associate Professor of Computer Science and Information security in the Department of Computer Science, Babcock University, Nigeria. Holds a Doctorate Degree from the Department of Computer Science, Babcock University, Nigeria. He can be contacted at email: agbajem@babcock.edu.ng.







**Theophilus A. Enem**     holds a PhD in Computer Science from Babcock University, Ilisan Remo, Ogun State, Nigeria. He has several years' experience of teaching computer science and cyber security courses at the university level. Currently, the HOD of Cyber Security Department, Air Force Institute of Technology, Kaduna, Nigeria. Enem is a full member of the Nigeria Computer Society (NCS) and the Computer Professional Registration Council of Nigeria (CPN). His areas of interest are Net-working, Telecommunications, and Forensic Science. He has published works in several journals of international repute. He can be contacted at email: ta.enem@afit.edu.ng.



**Lukman Adebayo Ogundele**     obtained his Ph.D. (in Computer Science) from Federal University of Technology, Akure, Nigeria. Currently, He is a Lecturer in the Department of Mathematical Sciences (Computer Science Unit), Olabisi Onabanjo University, Ago-Iwoye, Nigeria. His research interests are cyber security and information technology policy. He can be contacted at email: ogundele.lukman@oouagoiwoye.edu.ng.



**Suleiman A. Usman**     holds an MSc in Information Communication Technology from the National Open University, Kaduna, Nigeria. He has several years' experience of teaching Computer Science and cyber security courses at the Airforce Institute of Technology, Kaduna and has previously worked with Budget and Planning Commission as a Principal Data Processing Officer. He is a full member of the Nigeria Computer Society (NCS). His areas of interest are Machine Learning, Cyber Security, and Software Engineering. He can be contacted at email: usman.sule@afit.edu.ng.