AENSI Journals

Australian Journal of Basic and Applied Sciences

ISSN:1991-8178

Journal home page: www.ajbasweb.com

An Efficient AQM for Reliable Communication in Internet

[1]M. Danasekar and [2]Dr.V. Venkatachalam

[1]*Assistant Professor, Department of Computer Applications, Erode Sengunthar Engineering College, Thudupathi, Erode, Tamil Nadu, India*
[2]*Principal, The Kavery Engineering College, Mechery, Salem Dt., TamilNadu, India*

ARTICLE INFO

Article history:
Received 25 October 2014
Received in revised form
26 November 2014
Accepted 29 December 2014
Available online 15 January 2015

Keywords:
AQM

ABSTRACT

In today's Internet traffic packet loss due to congestion is the most important issue. An Active Queue Management (AQM) algorithm can reduce the packet loss in the Internet. AQM works at routers to control the congestion, where the packets may be dropped before queue become full. In this paper An Efficient AQM algorithm is implemented to overcome loss of the packet and provide high throughput for reliable communication in an Internet. Packet Loss can be reduced by tuning the arrival rate of the incoming packets based on queue size in receiver side. The main objective of an Efficient AQM algorithm is to control the flow rate, low packet loss as well as indicating status of the receiver queue size using QS bit. On receiving the QS bit, sender has to control the sending rate in order to avoid the packet loss.

To Cite This Article: M. Danasekar and Dr.V. Venkatachalam., An efficient AQM for Reliable Communication in Internet. ***Aust. J. Basic & Appl. Sci.,*** *9(2): 311-322, 2015*

INTRODUCTION

Congestion:

In data networking congestion occurs when a link or a node carries so much of data so that its quality of service may be reduced. In other words congestion may occur when the rate of incoming packet is more than the rate of outgoing packets. Congestion collapse generally occurs at choke points in the network, where the total incoming traffic to a node exceeds the outgoing bandwidth. When congestion occurs there will be less available throughput, high levels of packet delay and loss. When the sending rate is too high that could not be handled by intermediate routers, the intermediate routers discarded many packets, expecting the end points of the network to retransmit the information. Quality of service is extremely poor when congestion occurs in network. Congestion was first observed in the early Internet in October 1986. Due to this congestion collapse there are problems such as queue delay, less throughput and packet loss

Congestion control:

Congestion control refers to technique that can either prevent congestion before it occurs or remove congestion after it has happened. Basically there are two types of congestion control mechanisms. They are: Open-loop congestion control and Closed-loop congestion control.

Open-loop congestion control:

In open-loop congestion control, there are policies which were applied to prevent the congestion before it happens. The various policies are retransmission policy, window policy, acknowledgement policy, discarding policy, admission policy.

Closed-loop congestion control:

Closed-loop congestion control provides various mechanisms which try to lessen congestion after it happens. Some of the closed-loop mechanisms are backpressure, choke packet, implicit signaling and explicit signaling.

Transmission Control Protocol (TCP) uses congestion control to avoid or to remove congestion. One of the algorithms used in TCP congestion control is called slow start. This algorithm is based on the idea that the size of the congestion window starts with one maximum segment size (MSS). In slow start algorithm, the size of the congestion window increases exponentially until it reaches a threshold. As the name implies, the window starts slowly, but it grows exponentially.

Corresponding Author: M. Danasekar, Assistant Professor, Department of Computer Applications, Erode Sengunthar Engineering College, Thudupathi, Erode – 638 057, TamilNadu, India.
E-mail: danasekar.m@gmail.com

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

Congestion control techniques used to improve Quality of Service (QoS) which is main issue in the Internet. There are various techniques used to improve QoS. They are: Scheduling, traffic shaping, admission control and resource reservation. Scheduling techniques are designed to improve QoS, it includes FIFO queuing, priority queuing and weighted fair queuing. In first-in, first-out (FIFO) queuing, packets wait in buffer (queue) until the router is ready to process them. If the average arrival rate is higher than the processing rate, the queue become full and the packets will be discarded. In priority queuing, packets are placed in two different queue based on priority. The packets placed in the highest-priority queue are processed first and the packets placed in the lowest-priority queue are processed last. A priority queue can provide better QoS than the FIFO queue. But in priority queue, if starvation occurs then the lowest-priority queue will never be processed. To overcome this problem Weighted Fair Queuing method may be used. In this technique, the packets still assigned to different queue but packets were processed in a round-robin fashion so that all the packets will be processed.

Queue Management:

There are two kinds of queue management algorithms: Passive and Active. The traditional (Passive) technique for managing router queues is to set a maximum length (in terms of packets) for each queue, accept packets for the queue until the maximum length is reached, then reject (drop) subsequent incoming packets until the queue decreases because a packet from the queue has been transmitted. This technique is known as "tail drop", since the packet that arrived most recently is dropped when queue is full.

Active Queue Management (AQM) algorithm is an algorithm which is used for TCP networks congestion control. AQM is used to avoid congestion before it occurs, it can reduce the packet loss in the Internet. AQM effectively works at the router where the packets are dropped before the queue become full. There are numerous AQM algorithms were introduced to avoid congestion. They are: Random Early Detection (RED) algorithm, Adaptive RED algorithm, BLUE AQM algorithm, NEWQUE AQM algorithm etc., Each algorithm provides its own level of high throughput, average queue delay, low packet loss. In this paper we evaluate the performance of each algorithm based on stated parameters and we implement an Efficient AQM algorithm to provide a reliable communication in Internet.

MATERIALS AND METHODS

The following existing algorithms have been analyzed for proposing an efficient AQM algorithm

RED Algorithm:

RED algorithm (Sally Floyd, 1993) is the base of AQM algorithms and it is also called as congestion avoidance algorithm. In this algorithm queue size is configured manually based on the traffic in Internet.

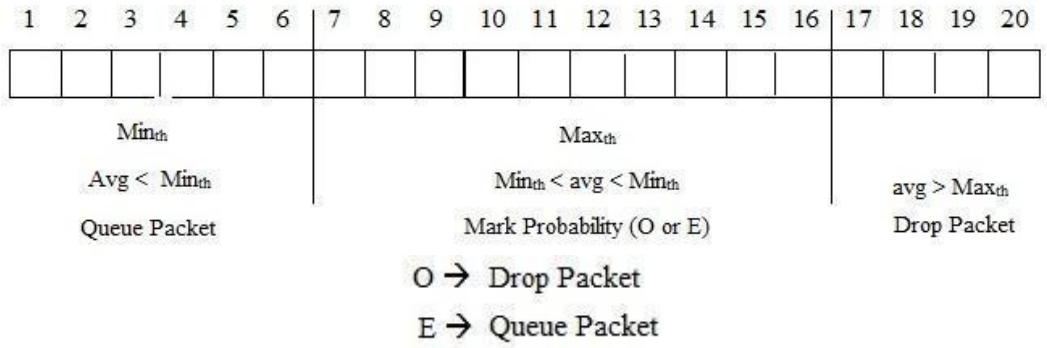


Fig. 2.1.1: RED Algorithm.

Average queue size is calculated using the formula

$$\text{Average queue size} = \frac{\min_{th} + \max_{th}}{2}$$

Where,

$\min_{th}$ minimum threshold

$\max_{th}$ maximum threshold

Average queue size is compared to two thresholds, a minimum threshold and a maximum threshold. When the average queue size is less than minimum threshold, incoming packets were accepted and no packets were dropped. When the average queue size is greater than the maximum threshold, every arriving packet is marked in fact dropped. If the average queue size is between minimum and maximum threshold, then each incoming packets will be marked either as optional or essential. Un-fragmented packets may be marked optional.

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

Fragmented packets may be marked as essential because loss of individual packet will not provide complete information to the receiver. When the queue length exceeds maximum threshold, then all packets which were marked as optional will be dropped so that congestion is avoided and at the same time packet delay can be reduced. In RED algorithm all the queue settings was done manually at router by the administrator.

Drawbacks:

In RED algorithm maximum queue size varies with the level of congestion. Queue size is tuned manually by the administrator. It is not fixed since congestion cannot be predicted by user or administrator. Due to this problem there is average queue delay at the gateway.

Adaptive RED algorithm:

The main goal of Adaptive RED algorithm (Sally Floyd, 2001) is to solve the problem of RED algorithm with minimal changes. In this algorithm queue length is tuned automatically. Average queue size is calculated using the formula,

$$\text{Average queue size} = \frac{\min_{th} + \max_{th}}{2}$$

Where,

$\min_{th}$ minimum threshold, 1% of Queue size

$\max_{th}$ maximum threshold, 50% of queue size

In Adaptive RED algorithm minimum threshold will be 1% of queue size considered as $\min_{th}$ and maximum threshold will be 50% of queue size considered as $\max_{th.}$ Adaptive RED, showing that by adapting $\max_p$ to keep the target queue size within the target range between $\min_{th}$ and $\max_{th}$, it is possible to achieve the same performance of that from RED with the value of maxp. Maximum Probability, Max_p defines the probability of packet drop. Based on the value of maximum probability, packets may be dropped. Two default parameters α and β is used to calculate Max_p. Here 'α' is the increasing factor and 'β' is the decreasing factor. However Adaptive RED, leaves the choice of the target queue size to network operators who must make policy tradeoff between utilization and delay.

Drawbacks:

In Adaptive RED algorithm, queuing delay may be occurred. Here target queue size is left to network operators.

BLUE AQM algorithm:

Fundamentally different active queue management algorithm is called BLUE (Wu-chang,2004); it uses packet loss and link idle to manage the congestion. The key idea behind this BLUE algorithm is to perform queue management based directly on packet loss and link utilization rather than on the instantaneous or average queue lengths.

BLUE algorithm maintains a single probability, Pm, which is used to mark the packets which are to be enqueued. If the queue is continually dropping packets due to buffer overflow, BLUE increments Pm, thus increasing the rate at which it sends back congestion notification. If the queue becomes empty or if the link is idle, BLUE decreases marking probability, Pm. BLUE uses other two parameters which control how quickly the marking probability changes over time. First parameter is freeze_time, it determines the minimum time interval between two successive updates of Pm. As per the author's statement(Wu-chang,2004), here the freeze_time is fixed as constant, this value should be randomized in order to avoid global synchronization. The freeze_time should be based on the effective round-trip times of connections multiplexed across the link in order to allow any changes in the Pm. The other parameters used (δ_1 and δ_2) determine the amount by which Pm, is incremented or decremented. The value of Pm is incremented by δ_1, when the queue overflows and the value of Pm is decremented by δ_2, when the link is idle.

NEWQUE AQM algorithm:

A NEWQUE AQM algorithm is generated for Explicit Congestion Notification (ECN) (Santhi.V ,2011).

Objective:

The main objective of NEWQUE AQM algorithm is to improve performance of congested router, to low average queuing delay, to provide low percentage of packet loss and to provide high throughput.

NEWQUE AQM algorithm works well for TCP versions such as TCP Vegas and TCP Newreno. It works for different scenario like short-lived TCP flow, long-lived TCP flow and also mixes both. In comparative study of various algorithm (Santhi.V ,2011) such as Random Early Detection (RED) algorithm, Adaptive RED

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

(ARED) algorithm, Random Exponential Marking (REM) and PI. This algorithm uses total flow arrival rate and link capacity. It maintains single marking probability. If arrival rate is greater than or equal to link capacity, then dropping probability will be increased. If arrival rate is less than or equal to link capacity then probability will be decreased.

Parameters Used:

Various parameter used in NEWQUE AQM algorithm are

B Queue size

Q Total queue length of all active flow

C Link capacity

$r_{new}(t)$ current total arrival rate at router at time, t.

NEWQUE algorithm uses total arrival rate and link utilization to manage the congestion. New arrival rate can be calculated using fixed weight exponential average method using the formula

New Arrival Rate is: $r_{new}(t) = (1 - e^{-T/K})\ 1 / T + e^{-T/K}\ r_{old}(t)$

Where, K constant

T inter arrival time, which is time between previous and current packet arrival.

NEWQUE algorithm [1] concludes that it maintains low loss rate, low average delay and high throughput.

An Efficient AQM Algorithm:

An Efficient AQM algorithm is implemented to overcome the packet loss and improve the throughput for reliable communication in an Internet. Packet Loss shall be reduced by tuning the arrival rate of the incoming packets based on queue size at the receiver end. The main objective of an Efficient AQM algorithm is to control the flow rate with minimum packet loss as well as indicating status of the receiver queue size using QS bit. On receiving the QS bit, sender has to control the sending rate in order to avoid the packet loss.

In existing AQM algorithms, RED Algorithm two threshold limits were considered Min and Max. When queue size exceeds Max threshold, dropping probability would be considered, where user has to specify optional/essential packet. Main drawback here was, each sender would consider their packets as essential one so no drastic change might occur due to dropping probability. In proposed algorithm three threshold limits are considered.

In NEWQUE AQM, queue capacity had been considered in which dropping probability was decided. If queue exceeds queue capacity, dropping probability would be incremented and if queue decreased below queue capacity, dropping probability would be decremented. Here ECN bit indicates the sender about the congestion at receiver end.

In all the above said existing algorithms, queue status was not informed to the sender. In proposed algorithm, queue status will be informed to each sender by means of QS bit. Based on QS bit, each sender will tune their sending rate in order to avoid packet loss.

Queue status has been informed to sender to avoid packet loss due to queue full. To avoid packet loss or congestion, senders have to consider queue status which is indicated using bit QS. Initially QS is set to -1, at this stage each sender will send n packets/min. If Que exceeds minimum threshold (min) then QS=0 will be send for each sender indicating ¼ th of queue is filled, so that each sender has to send n/2 packets/min to avoid packet loss. If Que exceeds maximum threshold (max) then QS=1 will be send for each sender indicating ½ th of queue is filled, so that each sender has to send only n/4 packets/min to avoid packet loss. When Que exceeds maxp, packets may be dropped based of dropping probability.

Diagrammatic representation of an EAQM Algorithm:

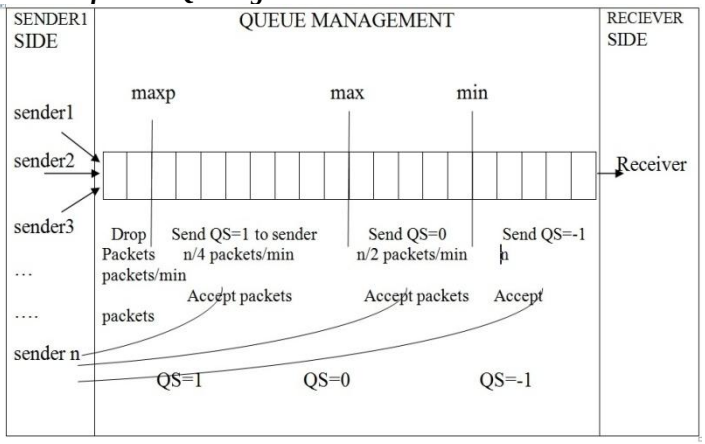


Fig. An EAQM Algorithm.

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

Arrival_rate = n packet/min
QS=Queue Status
Que Queue size filled with data
QS =-1 n packets/min (queue size < 25%)
QS=0 n/2 packets/min (25% < queue size < 50%)
QS=1 n/4 packets/min (50% < queue size < 90%)

ALGORITHM

At Receiver Queue

```
Que  size of queue filled with data
QS  Queue Status
min, max, maxp  Threshold limit values
QS=-1 //initialization
arrival_rate=n packets/min;
int min=25,max=50,maxp=90,que;
if(Que<min)
{
      QS=-1; //send(QS=0) to all sender;
      new_arrival_rate=arrival_rate;
      Accept the packet;
}
else if(min<Que<max)
{
      QS=0; //send(QS=0) to all sender;
      new_arrival_rate=arrival_rate/2;
      Accept the packet
}
else if(max<Que<maxp)
{
      QS=1; //send(QS=1) to all sender;
      new_arrival_rate=arrival_rate/4;
      Accept the packet
}
else if(Que>maxp)
{
      Mark packet dropping probability;
}
```

Sender Side

```
QS=-1;
send_rate=n packets/min;
if(QS==-1)
      new_send_rate=send_rate;
          //send n packets/min (eg: 100 packets/min)
else if(QS==0)
      new_send_rate=send_rate/2;
          //send n/2 packets/min (eg: 50 packets/min)
else if(QS==1)
      new_send_rate=send_rate/4;
         //send n/4 packets/min (eg: 25 packets/min)
else
      Mark packet dropping probability
```

Implementation:

Implementation of Existing RED algorithm

- RED algorithm has been implemented and its performance was measured using various parameters.
- In RED algorithm,
- Queue size was fixed
  Packet loss was too high, when all the incoming packets were marked as optional

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

In RED Algorithm Average queue size was compared with two thresholds, minimum threshold and maximum threshold. When the average queue size was less than minimum threshold, incoming packets were accepted and no packets were dropped. When the average queue size was greater than the maximum threshold, all the incoming packets were dropped. If the average queue size was between minimum and maximum threshold, then each incoming packets would be marked either as optional or essential. Unfragmented packets were marked optional. Fragmented packets were marked as essential because loss of individual packet will not provide complete information to the receiver. When the queue length exceeds maximum threshold, then all packets which were marked as optional would be dropped so that congestion was avoided and packet delay could be reduced.

RED Algorithm when avg<min

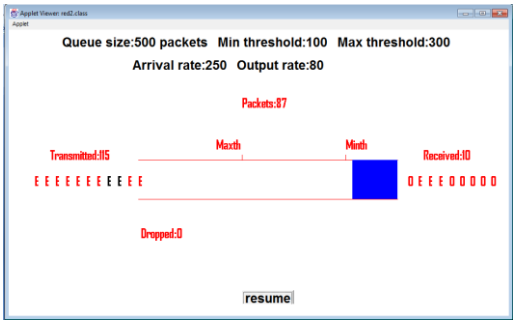


Fig. RED Algorithm when avg<min

RED Algorithm when min<avg<max

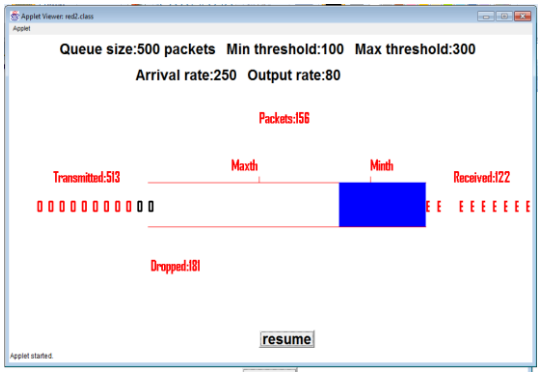


Fig. RED Algorithm when min<avg<max

In RED algorithm,

All the optional packets were dropped when queue size is between the threshold Min & Max. Here optional packets were lost. Here fragmented packets would be considered as essential and unfragmented packets were marked as optional. But most of the users would mark all their packets as an essential. Hence receiver queue would be filled soon and packets may be dropped. This algorithm had been implemented and analyzed to propose an efficient AQM algorithm

Implementation of EAQM algorithm:

In this paper, An Efficient AQM algorithm has been implemented. Receiver queue has been fixed with particular size. Three threshold values were assigned to the queue, such that minimum threshold as min, middle threshold value as max, and maximum threshold as max_p. Based on which sender should tune its sending rate. QS bit is used to indicate the Queue Status.

Flow of an EAQM algorithm is depicted below

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

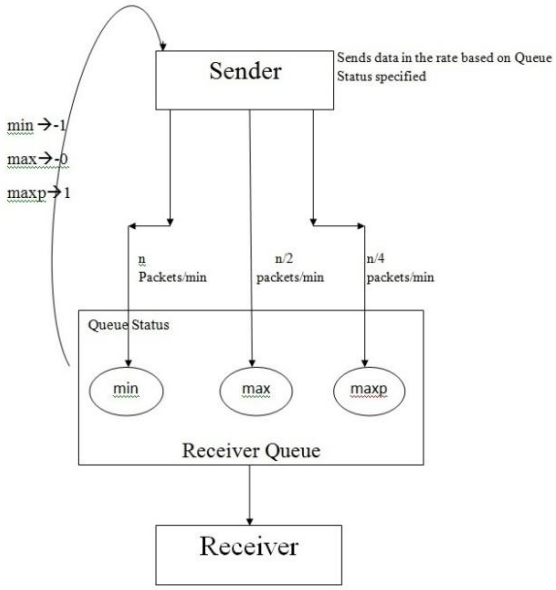


Fig. 5.6: Dataflow Diagram.

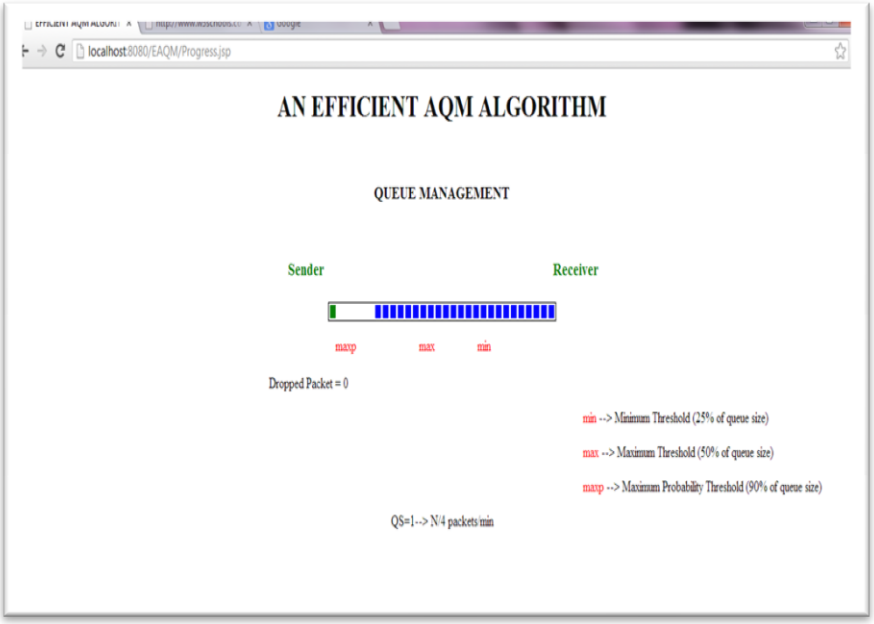
Initially QS is set to -1, at this stage each sender will send n packets/min. For example, 100 packets/min. When QS bit is set to -1, sender come to know that queue size is less than min threshold. Hence sender may send n packets per min.

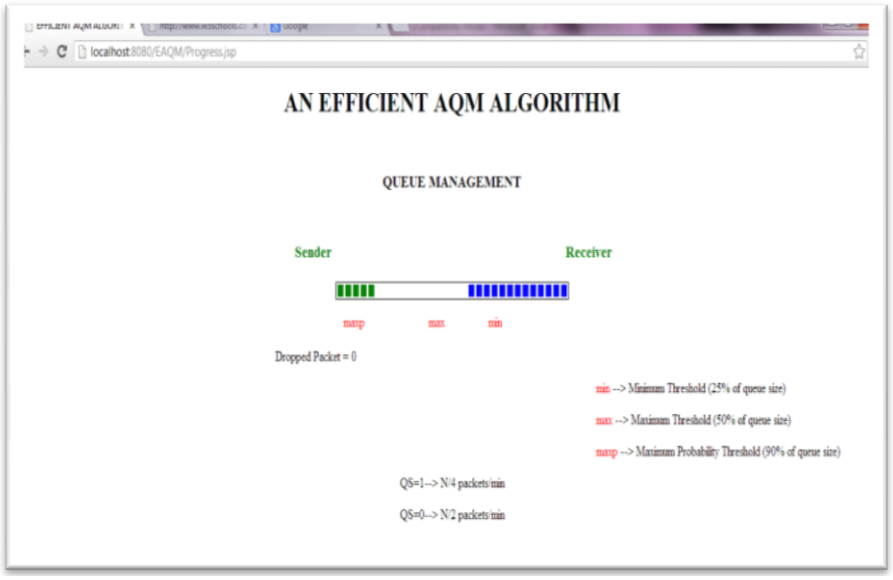
If Que exceeds minimum threshold (min) then QS=0 will be send for each sender indicating ¼ th of queue is filled, so that each sender has to send n/2 packets/min to avoid packet loss.

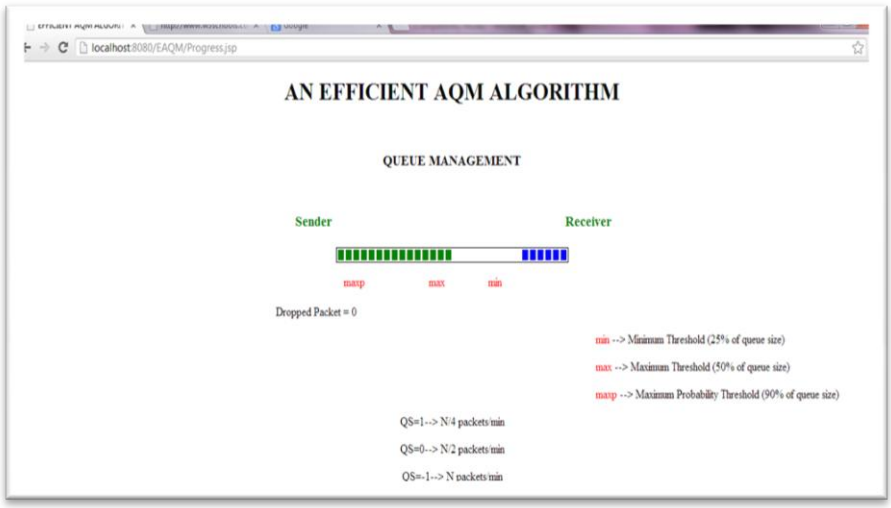
If Que exceeds maximum threshold (max) then QS=1 will be send for each sender indicating ½ th of queue is filled, so that each sender has to send only n/4 packets/min to avoid packet loss.

When Que exceeds max_p, packets may be dropped based of dropping probability. Here dropping probability will be marked by user, like RED algorithm. Optional / Essential packets should be marked by the user, where optional packets may be dropped.

Output Screen of Eaqm Alogithm:







RESULTS AND DISCUSSIONS

Performance of existing RED algorithm and proposed EAQM algorithm had been analyzed using three parameters.

1. Automatic Queue Tuning
2. Packet loss
3. Throughput

Automatic Queue Tuning:

Queue tuning is achieved by the router automatically using the proposed algorithm. Here arrival rate has been reduced based on the queue size hence there is no packet loss due to this automatic tuning. Automatic tuning had been proven using the flow of data in the proposed algorithm.

Packet Loss:

Packet loss has been analyzed for both the RED and EAQM algorithm, comparative analysis was carried out to prove EAQM achieve its objective i.e., Low packet loss

The above analysis depicts that, packet loss is very low in the proposed Efficient AQM algorithm than existing RED algorithm. In existing algorithm when threshold limit exceeds min, optional packets were dropped. There occurs loss of packet when queue size exceeds threshold limit. But in proposed EAQM algorithm, there is no packet loss in any of the threshold limit because arrival rate is tuned accordingly.

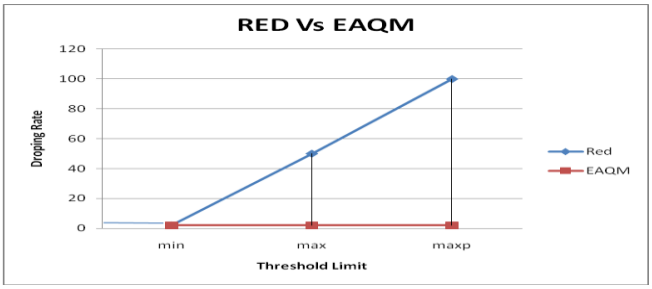


Fig. 5.4: Packet Loss.

Throughput:

Throughput is the number of packets received per minute. Here performance analysis for throughput had been done.

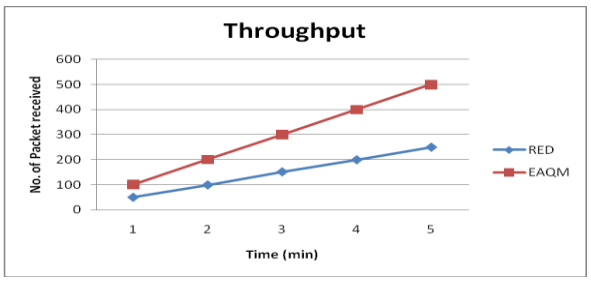


Fig. 5.5: Throughput.

In the existing algorithm i.e. RED, number of packets received per min less than the number of packets sent due to packet loss. Here in proposed algorithm, number of packets received is same as numbers of packets send. Hence the above analysis depicts, best throughput had been achieved in proposed algorithm than the existing RED algorithm.

Conclusion & Future Enhancement:

Conclusion:

In this paper, An Efficient AQM algorithm is a queue management algorithm, where queue status is informed to sender. It will tune the queue automatically. Sender will tune the sending rate based on bit received, so that there is no packet loss. Proposed algorithm provides low packet loss, since sender reduces the sending rate when queue become filled. At the same time, sending rate will be increased when queue become empty. Hence an efficient AQM provides best throughput and low packet loss.

Future enhancement:

When the threshold limit exceeds the max_p, free space available in the router may be utilized to hold optional packets instead of dropping the incoming packets. Optional packets which were stored in the router may be sent to the receiver when queue become free. Solution may be obtained for the case where QS bit has been lost.

REFERENCES

Abdullah AI Musud, Hossain Md. Shamim, Amina Akther, 2011. Performance Analysis of AQM Schemes in Wired and Wireless Networks based on TCP flow. International Journal of Soft Computing and Engineering, 1.

Dan Shyan Hwang.I, Bor-Jiunn Hwang, Pen-Ming Chang, Cheng-Yu Wang, 2010. QoS-Aware Active Queue Management for Multimedia Services over the Internet. Proceedings of International Multi Conference of Engineers and Computer Scientists, Vol.2.

Dongyu Qiu, 2010. On the QoS of IPTV and its Effects on Home Networks. International Journal of Digital Multimedia Broadcasting.

Mohamed Hanini, Said El Kafhali, Abdelkrim Haqiq, Amine Berqia, 2011. Effect of the Feedback Function on the QoS in a Time Space Priority with Active Queue Management. International Journal of Computer Applications, 2.

Olawoyin, L.A., N. Faruk, L.A. Akanbi, 2011. Queue Management in Network Performance Analysis. International Journal of Science and Technology, 1(6).

Sally Floyd, Ramakrishna Gummadi and Scott Shenkar, 2001. Adaptive RED: An Algorithm for increasing the Robustness of RED's Active Queue Management.

Australian Journal of Basic and Applied Sciences, 9(2) February 2015, Pages: 311-322

Sally Floyd, Van Jacobson, 1993. Random Early Detection Gateways for Congestion Avoidance".

Santhi, V., A.M. Natarajan, 2010. MNEWQUE: A New Approach to TCP / AQM with ECN. International Journal of Computer Application, 3(3).

Santhi, V., A.M. Natarajan, 2011. Active Queue Management Algorithm for TCP Networks Congestion Control. European Journal of Scientific Research, 54(2).

Santhi, V., A.M. Natarajan, 2010. Performance Improvement on the Robustness of TCP / Active Queue Management with ECN. IACSIT International Journal of Engineering and Technology, 2(3).

Wu-chang Feng, Kang G. Shin, Fellow, IEEE, Dilip D. Kandlur, Debanjan Saha, 2002. The BLUE Active Queue Management Algorithms. IEEE transactions on networking, 10(4).