

# A NEW HISTORICAL BASED POLICING ALGORITHM FOR IP NETWORKS

Tarek M. Heggi<sup>1</sup>, Sherine M. Abd El-Kader<sup>2</sup>, Hussein S. Eissa<sup>3</sup>, Hoda A. Baraka<sup>4</sup>

<sup>1</sup>Assistant Researcher, Agricultural Research Center, Software & Hardware Maintenance Dept., Cairo, Egypt

<sup>2,3</sup>Associate Professor, Electronics Research Institute, Computers and Systems Dept., Cairo, Egypt

<sup>4</sup>Professor, Faculty of Engineering, Computer Engineering Dept., Cairo University, Egypt

## ABSTRACT

Development of QoS approaches has become mandatory to comply with the requirements of the new applications. Delay, loss, and jitter are the major and essential constraints to provide QoS, for example packet loss will cause chop and skips effects for voice traffic and also cause glitches and cutouts problems for video traffic. There are two policing algorithms developed by Cisco called Committed Access Rate (CAR) and Class-Based (CB). CAR is considered as a legacy approach to police traffic whereas CB is a newer configuration which is recommended by Cisco to be used for policing. This paper proposes a new policing algorithm; called Historical Based Token Bucket (HTB) algorithm. This paper also studies the impact of deploying HTB algorithm on real time traffic from the delay and losses point of view. The results of this paper concluded that the HTB algorithm reduces the losses by on average 72% and 99% less than the CB algorithm for different types of video and voice respectively, whereas the HTB algorithm increases the delay by about 4% and 9% more than the CB algorithm for different types of video and voice respectively.

**Keywords:** Committed Access Rate, Class Based, Token Bucket, Integrated Services, Differentiated Services and Quality of Services.

## 1 INTRODUCTION

The startling growth of the Internet and the flexibility of the IP-based packet switched networks have expedited the convergence of data, voice and video communications into single IP-based core architecture [1].

Development of QoS mechanisms are required to provide different levels of performance assurance, guaranteed bandwidth and packet delivery within specific constraints to face scaling problems in the Internet infrastructure such as: the growth of the Internet users, growth in heterogeneity, and the distributed Internet administration.

The process of managing the network resources is known as Traffic Engineering (TE), which concerned with performance optimization for the networks [2] and mapping traffic flow onto the physical topology to enhance the overall network utilization and create a uniform distribution for the traffic throughout the network [3]. There are a lot of organizations that have low speed links, so, this paper try to provide them with the best usage solution for their bandwidth through applying HTB algorithm in the edge of their network. The simplest and most obvious technique to

provide high service level without congestion is by over-provisioning the network, but this approach has several disadvantages because it is not economical to build the network from outset that is designed to cope with all types of traffic. Each organization needs to build her network based on its needs, to reduce the upfront capital investment and the resulting risk; also without the QoS capabilities, it is impossible for a service provider to offer service differentiation to customers, where services provided are linked to the amount paid for this service.

Integrated Services (IntServ) and Differentiated Services (DiffServ) are the most popular QoS models over the Internet. The IntServ is a QoS architecture that works on a small-scale network to determine the required level of service for each application [4]. The IntServ model includes two types of service targeted towards real-time traffic; guaranteed and controlled-load services. The DiffServ is an architecture that specifies a simple, scalable and coarse-grained mechanism for classifying, managing network traffic and providing QoS guarantees on different scales of IP networks [4]. Every class of traffic enters the boundaries of the DiffServ network is assigned to different behavior aggregates at the core routers to

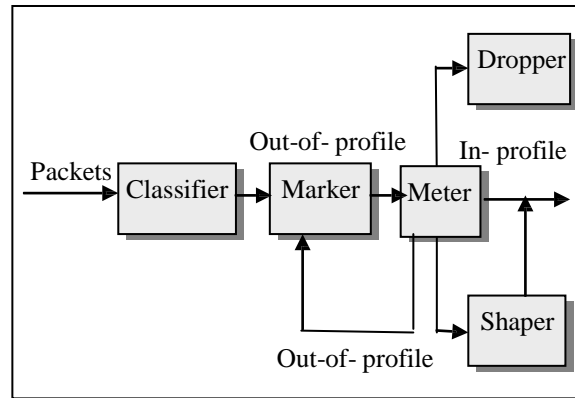
provide service differentiation for the aggregate traffic. Each behavior aggregate is identified by a single DiffServ Code Point (DSCP). Within the core of the network, packets are forwarded according to the per-hop behavior associated with that DSCP code point [5]. The IntServ and DiffServ models are working in the transport layer and they are based on different components such as classifier, marking, shaping and policing. One of the essential

models is the traffic policing technique. Policing is the QoS component that limits traffic flow to a configured bit rate, with limited bursting capability, packets above the specified burst rate are dropped or have their precedence altered [6].

This paper proposes a new traffic policing algorithm based on a single token bucket with two variables for the token generation rate and the bucket size length which are mainly based on the characteristics of the online traffic. The rest of the paper is organized as follows. Section 2 discusses the different QoS IP models and presents some effort that had been done to enhance the service level of the real time traffic. Section 3 describes the traffic policing approach and the operation of the token bucket algorithm. Section 4 presents the new HTB algorithm. Section 5 discusses the obtained results. Finally, section 6 summaries and concludes the paper.

## 2 QOS IN IP NETWORK

QoS is a mechanism to assure that the traffic which traverse the Internet such as audio and video data; will have minimum delay, loss and high throughput. If QoS is not supported, the IP voice or videoconferencing calls will be unreliable, inconsistent, and often unsatisfactory. Customers and service providers have different requirements for their QoS and it is interpreted in different ways. For example, QoS to service providers means the ability to offer different grades of service so that they can charge their customers differently. The underlying factor for service providers is lower costs and increased revenue. In the case of customer, QoS has other implications; it means good management for their links, improvement in application response time and priority in handling data traffic as a matter of policy or resource. To implement the QoS policies, the network hardware as the routers should have the capability for traffic conditioners. Traffic conditioners mean classifiers, meters, markers, shapers, and droppers [7]. Traffic conditioners are usually located within DiffServ boundary nodes i.e., ingress and egress nodes. Figure (1) illustrates the traffic conditioners. The classifier selects packets in a traffic stream based on the content of some portion of the packet header. The marker sets the DSCP value of each packet; effectively mark it to a particular behavior aggregate.



**Figure (1):** The QoS Components in the IP router

The meter measures the temporal properties of the stream of packets selected by a classifier against a traffic profile specified in the Service Level Agreement (SLA) offered by the domain. A meter passes state information to other conditioning functions to trigger a particular action for each packet which is either in- or out-of-profile. Shaper delays some or all of the packets in a traffic stream in order to bring the stream into compliance with a traffic profile. A shaper usually has a finite-size buffer, and packets may be discarded if there is not sufficient buffer space to hold the delayed packets. Dropper discards some or all of the packets in a traffic stream in order to bring the stream into compliance with a traffic profile. This process is known as "policing" the stream. Note that a dropper can be implemented as a special case of a shaper by setting the shaper buffer size to zero (or a few) packets.

These QoS components are used in the two board QoS categories, fine-grained which is represented in IntServ and coarse-grained which is represented in DiffServ [8].

### 2.1 Integrated Services

The IntServ essentially defines an end-to-end pathway through the network for each application's packets, which is similar to the Asynchronous Transfer Mode (ATM). IntServ model provides a general framework that allows IP-based applications to use multiple quality levels for their traffic flows [9]. Resource Reservation Protocol (RSVP) was designed by the IETF as one of the basic components of the IntServ model. RSVP supports two service classes: guaranteed service class for applications with fixed delay bound and controlled-load service for application that require reliable and enhanced best-effort service. In the guaranteed service class, routers forward packets strictly within the delay bounds specified by the receiver. The controlled-load service provides a QoS level equivalent to a best-effort service under unloaded or moderately loaded

network conditions. RSVP soft state is created and periodically refreshed by Path and Resv messages. The Path message contains the specifications of data traffic when the Path message eventually reaches the destination end-points the receivers may choose to send Resv message to the network, in the direction of the sender contains the desired QoS. The main drawback of the IntServ model is that both of the flow classes and the reservation techniques is not scalable due to the expense of high signaling load resulting in the network [10].

## 2.2 Differentiated Services

DiffServ model tries to solve the scalability problem of IntServ structure by separating the role of boundary nodes from that of the core nodes [11]. DiffServ alleviates the complexity in core routers by assigning all complex processing such as flows classification and traffic dimensioning to edge routers. In the backbone, forwarding Per-Hop Behaviors (PHB) are defined, namely, Expedited Forwarding (EF) and Assured-Forwarding (AF) [12]. PHB defines the policy and priority applied to a packet when traversing a hop in a DiffServ network. EF PHB is to provide a building block for low loss, low delay, and low jitter services. AF proposes simple mark and drop mechanisms to realize IP QoS and provides better than best-effort service by controlling the drop preference of the packets at the time of congestion. DiffServ model solves the scalability problem in the core network therefore, it is better for the application than IntServ model, but it cannot provide perfect QoS because it doesn't support the function to guarantee QoS for each flow.

## 2.3 Related Work

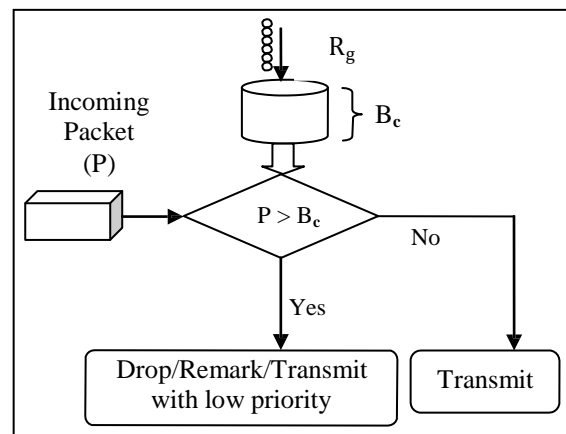
Paper [13] proposes an adaptive token bucket algorithm for achieving proportional sharing of bandwidth among aggregate flows in DiffServ networks. The author of the paper states that the aggregate flow with a lower target rate occupies more bandwidth than its fair share, while the aggregate flow with a higher target rate gets less than its fair share. So this paper proposed an algorithm that solves this unfairness problem by adjusting the target rate according to the edge-to-edge feedback information. But unfortunately, this proposed algorithm only supports the TCP traffic. In [14] an algorithm has been adapted to assign the optimal values for tokens to the individual long live TCP flows in DiffServ networks. The optimization process inside the algorithm is based on Genetic Algorithm (GA) approach which was deployed using C++ simulator. The results of the simulation concluded that the values of tokens will be varied according to the incoming rate of application.

## 3 TRAFFIC POLICING APPROACH

Traffic policing meters network traffic for conformity with a traffic contract and if required, dropping traffic to enforce compliance with that contract. The following subsections demonstrate two mechanisms for traffic policing which based on the token bucket algorithm.

### 3.1 The Token Bucket Algorithm

Traffic policing elements comprise a meter and a dropper. They may also optionally include a marker. Token bucket is a common scheme used to measure the traffic. It is used in both the policing and shaping algorithms as a means to report whether a packet is compliant or noncompliant with the rate parameters configured for it [15]. Figure (2) shows a simple diagram for token bucket algorithm.



**Figure 2:** The Token Bucket Algorithm

The token bucket consists of a bucket with a maximum capacity of  $B_c$  tokens which refills at a rate  $R_g$  tokens per second, each token typically represents a quantity of whatever resource is being rate limited. The bucket contains tokens, each of which can represent a unit of bytes or a single packet of predetermined size. Token bucket main parameters are token arrival rate  $R_g$ , bucket depth  $B_c$ , and time interval  $T$ . The bucket can hold  $B_c$  tokens. If a token arrives when the bucket is full, it is discarded. When a packet of  $P$  bytes arrives, an equivalent number of tokens are removed from the bucket, and the packet is sent to the network. If less than the number of the available tokens, no tokens are removed from the bucket, and the packet is considered to be non-conformant. The algorithm allows bursts of up to  $B_c$  bytes, but over the long run the output of conformant packets is limited to the constant rate  $R_g$ . Non-conformant packets can be treated in various ways: dropped, queued for subsequent transmission when sufficient tokens are accumulated in the bucket, or marked as being non-

conformant, possibly to be dropped subsequently if the network is overloaded.

Any two values of the token bucket may be derived from the third by the relation presented in equation (1) [16].

$$T(\tau e) = \frac{B - \tau y e}{Rg - \tau e} \quad (1)$$

### 3.2 CAR and CB Policers

This paper pays attention to the policing techniques that are used in the traffic conditioner components. The two primary mechanisms for traffic policing in Cisco IOS are CAR and CB policers [17].

CAR has two QoS functions: bandwidth management through policing, and packet classification through IP precedence, QoS group, or IP access list. Classification in CAR is applied on interfaces for all IP traffic. There are two actions for conforming and non-conforming traffic which are named as follows “conform” and “exceed”. CAR uses a single token bucket for both normal and maximum burst. During the operation of traffic measurement in CAR, it replenishes bucket continuously every time interval by adding the size of conform tokens to the bucket. CB is implemented by using the modular QoS Command Line Interface (CLI) by configuring a service policy. There are three actions for conforming, non-conforming traffic, and violating traffic which are named as follows “conform”, “exceed”, and “violate”. CB uses two token buckets the first one for normal burst while the second one for the maximum burst. During the operation of traffic measurement in CB, it replenishes tokens in bucket in response to police a packet as opposed to every time interval in seconds.

The most significant functional difference between CAR and CB is the using of the two-bucket mechanism. The token bucket algorithm provides users with three actions for each packet: a conform action, an exceed action, and a violate action. Traffic entering the interface with traffic policing configured is placed into one of these categories. Within these three categories, users can decide packet treatments. For instance, packets that conform can be configured to be transmitted; packets that exceed can be configured to be sent with a decreased priority; and packets that violate can be configured to be dropped therefore, Cisco recommended the using of CB and the avoiding of using CAR, for which no new features or functionality is planned.

## 4 THE HISTORICAL BASED TOKEN BUCKET ALGORITHM

The following subsections describe the operation sequence of the traffic policing approach and the design of the HTB algorithm

### 4.1 Traffic Policing Approach

The studied experiment is based on capturing and measuring different voice and video Internet traffic by using Wireshark measurement tool [18]

then classifying the traffic into different classes depend on their characteristic by using the HTB developed tool as shown in figure (3) which is based on the application port number. The HTB tool has been developed by using C Sharp (C#) to convert the collected raw captured voice and video traffic to Comma separated values (CSV) structured format, check the source IP that initiates the traffic to confirm that these packets belong to a specific source IP address and a specific streaming server, and to set the values of the class name, token bucket size, token generation rate, and the update interval depend on the classification process of the real time traffic. The HTB tool has been designed to study the discrepancies between both of the CB and HTB algorithms. It should be noted that the experiment is done for one hour and the measured traffic has been collected during the working hour for six months.

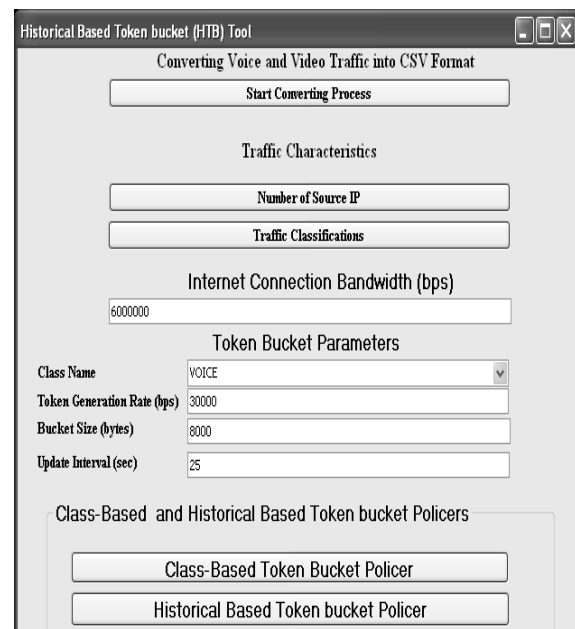
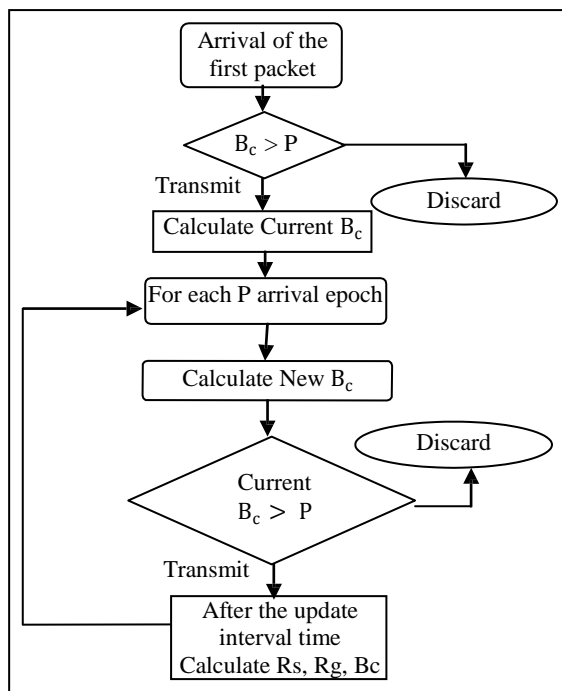


Figure 3: Historical Token Bucket Tool

### 4.2 Historical Based Token Bucket Algorithm

The new Historical based Token Bucket (HTB) algorithm uses a single token bucket algorithm, traffic should be transmitted based on the presence of

tokens in this bucket, each token represent a given number of bytes, which is assumed to be one byte in our implementation, and the bucket can hold up to  $B_c$  tokens. Traffic is allowed to transmit up to its peak burst rate if there are adequate tokens in the bucket and if the burst threshold is configured appropriately, figure (4) demonstrates the operation of the HTB algorithm. When a packet of  $p$  bytes arrives,  $p$  tokens are removed from the bucket, and the packet is allowed to be sending to the network. If less than  $p$  tokens are available in the bucket, no tokens are removed from the bucket, and the packet will be discarded. The main differences between the proposed HTB algorithm and the traditional token bucket which is used in the CB algorithm are: first, in the HTB algorithm the value of the token generation rate is changed every update interval depending on the traffic characteristic, the update interval also is varying from one second until it reaches its maximum at 25 seconds. Second, in the HTB algorithm the size of the token bucket is also changed depending on the traffic characteristic.



**Figure 4:** The Historical Based Token Bucket Algorithm.

At the beginning of the session the  $B_c$  start values in both CB and HTB algorithms are the same for the first iteration of the update interval. In the next iteration  $B_c$  is calculated based on the behavior of the voice and video traffic, as indicated in equation (1). At each iteration, the sender data rate  $R_s$ , the token generation rate ( $R_g$ ) and the token bucket depth ( $B_c$ ) will be recalculated based on the traffic characteristic variation, as shown in equations (1), (2) and (4).

The data rate of the previous interval time equals the total lengths of packets through this interval divided by the interval time.

$$R_s = \frac{N}{D} \tag{2}$$

Where:

$R_s$  is the sender data rate (bit per seconds),

$N$  is the total number of bytes that have been transmitted (bytes),

$D$  is the duration of the specified period (seconds), represent the value of the token generation rate.  $R_g$  is based on the amount of the available bandwidth  $C$ .

$$\begin{aligned} & i=n \\ \text{if } & R_s > C \\ & i=1 \end{aligned} \tag{3}$$

Where  $i$  is the total number of connections.

Then,

$$R_g = (P_d \times R_s) \tag{4}$$

Where,  $P_d$  is the ratio between the total number of data rate of the senders to the capacity of the Internet link.

$$P_d = \frac{TR_s}{C} \tag{5}$$

Where  $TR_s$  is the total data rate that consumed by all senders in the previous interval.

On the other hand

$$\begin{aligned} & i=n \\ \text{if } & R_s \leq C \\ & i=1 \end{aligned} \tag{6}$$

Then,

$$R_g = R_s \tag{7}$$

For each period both of the  $R_g$  and  $B_c$  values are calculated based on the historical data of the previous period and then they are used as a new values for the next period. The simulation study gradually increases the updating interval from 1 to 25 seconds to study the effect of modifying the update interval.

For the each packet, the tokens generation is calculated as the follows:

$$\frac{\text{New tokens} = (\text{the arrival time of the current packet} - \text{the arrival time of the new packet}) \times R_g}{8} \quad 8$$

The bucket size is calculated as follows:

$$B_c \text{ current} = B_c - P \quad (9)$$

For the voice traffic, the value of  $B_c$  is calculated as follows

$$B_c = P \quad (10)$$

For the video traffic, the value of  $B_c$  is calculated as follows

$$B_c = \sum P \quad (11)$$

## 5 RESULTS AND DISCUSSION

The following subsections illustrate the attribute of the traffic that has been collected, and results of both video and voice traces.

### 5.1 Characteristics of Captured Traffic

The main objective of this paper is to analyze the results of deploying the HTB and CB algorithms on various types of traffic to study the influence of applying the two algorithms on the losses and delay of the both of them. To achieve this objective, three different video traces and three different voice traces have been collected with taking into considerations that they have different characteristics. The first video trace is a news channel for Al Jazeera channel. The second video trace is a social program channel for El-Hayat channel. The third video trace is a sport channel for fighting channel. The first voice trace is classical world music for BBC radio channel. The second voice trace is British Broadcasting Corporation (BBC) radio news. The third voice trace is Jungle Drum and Pass channel for Jungle DNB radio channel.

As shown in table (1), the average packet sizes of the three video traces are 1194, 1381, and 1366 bytes respectively. The average packet sizes of three voice traces are 974, 697, and 1361 bytes respectively. The average data rates of three video traces are 62.6, 141.5, and 75.8 Kbps respectively. The average packet sizes of three voice traces are 67.9, 45.6, and 133.1 Kbps respectively. The number of packets in video traces ranges from 23601 to 46121 packets, on the other hand, it ranges from 29475 to 44014 packets in the voice traces. The above mentioned

parameters have been measured by using Wireshark network analyzer tool.

**Table 1:** The traffic characteristics for different traces.

| Trace Type   |                            | No. of Packets | Avg. Packet size (bytes) | Avg. data rate (bytes/sec) |
|--------------|----------------------------|----------------|--------------------------|----------------------------|
| Video Traces | Al Jazeera                 | 23601          | 1194                     | 7828                       |
|              | El-Hayat                   | 46121          | 1381                     | 17682                      |
|              | USA Fighting Sports        | 24969          | 1366                     | 9478                       |
| Voice Traces | BBC RADIO3 Classical Music | 31372          | 974                      | 8488                       |
|              | BBC RADIO5 News            | 29475          | 697                      | 5704                       |
|              | Jungle DNP Radio           | 44014          | 1361                     | 16643                      |

### 5.1 Effect of HTB Algorithm on Video Traffic

The effect of applying both of the HTB and CB algorithms on the performance of different video traffic is shown in figures (5) and (6). Figure (5) demonstrates that by applying the HTB algorithm on the Al-Jazeera, El-Hayat, and the Fighting video traffic, the number of lost bytes is reduced by approximately 84.7%, 99.9%, and 30.4% respectively than that obtained by applying the CB algorithm. Also it could be shown, that there is a variation in the amount of the dropped bytes from the first second to the twenty five second for all video traffic especially for the Fighting sports channel and this is due to the variation in the characteristics of that traffic during different period.

In Al-Jazeera video traffic as an example, the total number of bytes that transmitted over the network during one hour is approximately equals to 25 Mbytes, the CB algorithm dropped a fixed number of bytes which is 1.5 Mbytes during the entire duration whereas the HTB algorithm drop about 906.8 Kbytes. The HTB algorithm could decrease the number of dropped packet to about 307 Kbytes at the update interval of 25 seconds. It should be noted that at the first second of the learning period the amount of the dropped bytes in the HTB algorithm is very close to the CB algorithm and this is lead us to the fact that one second is not sufficient for the token bucket to obtain the accurate values that describes the behavior of traffic.

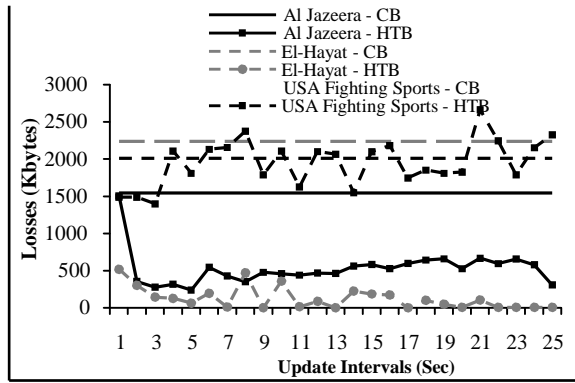


Figure 5: Losses of Video traffic for CB and HTB algorithms

Figure (6) illustrates the effect of applying the HTB and CB algorithms on the delay of the Al-Jazeera, El-Hayat, and the Fighting video traffic, the delay of the HTB algorithm is higher than the CB algorithm by only 5.5% for Al-Jazeera, and 3.7% for both El-Hayat and the Fighting video traffic.

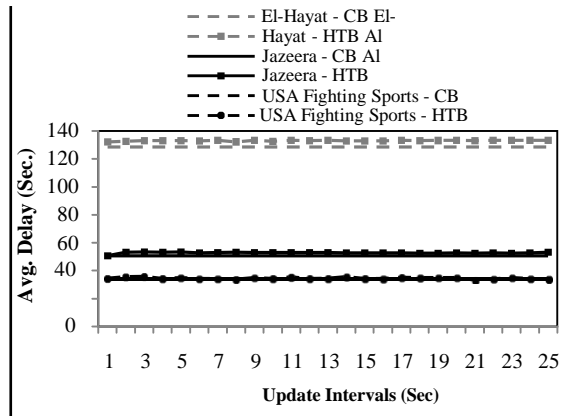


Figure 6: Delay of Video Traffic for CB and HTB algorithms

5.2 Effect of HTB Algorithm on Voice Traffic

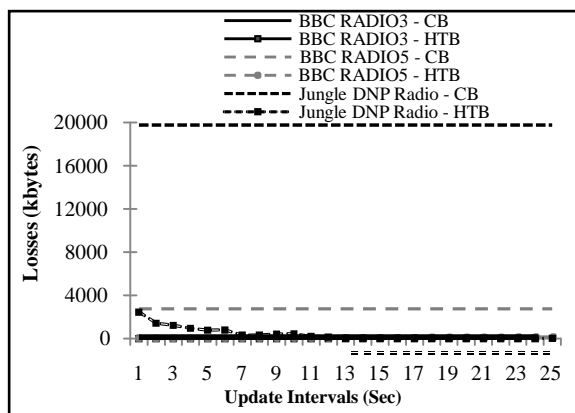


Figure 7: Losses of Voice Traffic for CB and HTB algorithms

Figure (7) illustrates the effect of applying both of the HTB and CB algorithms on the performance of different voice traffic. It could be shown that the HTB algorithm reduces the number of dropped bytes by on average 99.8% less than the CB algorithm for all studied voice traffic types. It should be clear that the total number of transmitted bytes for the BBC RADIO5 News is equals to 40 Mbytes, the CB algorithm dropped about 2.7 Mbytes whereas the HTB algorithm only dropped about 74 Kbytes, which means that the HTB algorithm studies well the behavior of the traffic during their transmitted time and this study lead to less losses percentage than that of the CB algorithm; which doesn't care about the characteristic of the traffic.

Figure (8) demonstrates that the delay of the HTB algorithm is higher than the delay of the CB algorithm by 6.2%, 7.4%, and 20% for the BBC RADIO3 Classical Music, BBC RADIO5 News and Jungle DNP Radio traffic respectively, and this is due to the delay resulting from the learning time. In Jungle DNB Radio as an example, the total number of bytes that assumed to be transmitted equals to 118Mbytes. In the CB algorithm the total number of dropped bytes is a fixed number equals to 19.7 Mbytes during the entire duration, in the HTB the number of dropped bytes are on average equals to 1.2 Mbytes.

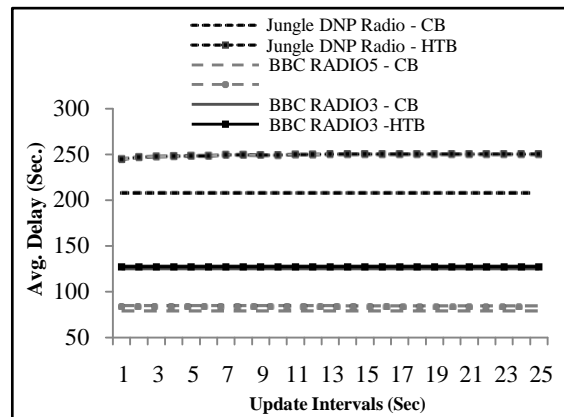


Figure 8: Delay of Voice Traffic for CB and HTB algorithms

6 CONCLUSION AND FUTURE WORK

This paper proposes a dynamic QoS approach for both the customers and the ISPs that provide a good utilization for their bandwidth. It studies the impact of deploying the HTB algorithm on different real time streams from losses and delay point of views. The paper also compares the proposed algorithm with the well known Cisco policing algorithm which is known as CB. The experimental results have lead to two conclusions, firstly, the losses in HTB algorithm is less than the CB algorithm by on

average 84.7% in case of Al-Jazeera video stream, 99.9% in case of El-Hayat video stream and 30.4% in case of US Fighting video stream whereas, the delay in HTB algorithm is more than the CB algorithm by 5.5% in case of Al-Jazeera video stream, 3.7% in case of El-Hayat video stream and the US Fighting video stream. The HTB algorithm reduces the losses up to 100% in case of BBC RADIO3 voice stream, 99.7% in case of BBC RADIO5 voice stream and the Jungle DNP Radio voice stream whereas, the delay in HTB algorithm is more than the CB algorithm by 6.2% in case of BBC RADIO3 voice stream, 7.4% in case of BBC RADIO5 voice stream and 20% in case of Jungle DNP Radio voice stream. Also it should be mentioned that the HTB algorithm does not require more processing than the CB and it an adaptive algorithm that can be suitable for several types of Internet traffic.

Secondly, the update interval, which is the online time that taken to study the characteristic of the traffic and to set the HTB parameters, is variable and depending on the traffic type. This paper recommend to set the value of the update interval to 25 seconds for the studied video traffic, this value decreases the losses value approximately to 72% and increases the delay by only 4% than the CB algorithm. Also it recommend to set the value of the update interval to 12 seconds for the studied voice traffic, this value decreases the losses value up to 99% and increases the delay by 9% than the CB algorithm.

In the future, more sophisticated classification technique which uses data mining algorithm is required to improve the accuracy of the HTB results.

## 7 REFERENCES

- [1] C. Chuah: Scalable Framework for IP-Network Resource Provisioning through Aggregation and Hierarchical Control, PhD thesis, University of California, (2001).
- [2] X. Xiao, A. Hannan, B. Bailey, and L.M. Ni: Traffic Engineering with MPLS in the Internet, IEEE Network Magazine, No. 14(2), pp. 28-33, (2000)
- [3] J. A. Zubairi and W. Al-Khateeb: MPLS Managing the New Internet, Bulletin of Institution of Engineers Malaysia, BIL, No.12, pp. 30-38 (2003).
- [4] J. Evans, C. Filstis: Deploying IP and MPLS QoS for Multiservice Networks Theory and Practice, Academic Press, (2007).
- [5] N. Zotos; G. Xilouris, E. Pallis, A. Kourtis: An MPLS-DiffServ experimental core network infrastructure for E2E QoS content delivery" Computer Systems and Applications, IEEE/ACS International Conference on Volume , Issue PP: 947 – 951 (2008).
- [6] P. Fouliras and C. Georgiadis: On the Reduction of Congestion for Multimedia Streaming in DiffServ Networks, Univ. of Macedonia, Thessaloniki, 5th International Conference on Technology and Automation (ICTA'05), Greece, pp. 312-317 (2005).
- [7] Cisco Inc: Deploying Cisco QoS For Enterprise Networks", Chapter 7, Policing and Shaping, published by Element K Content LLC, (2004)
- [8] A. Farrel, A. Gerald, D. Bruce, E. John: Network Quality of Services, Morgan Kufman, (2008).
- [9] M. Lee, K. Kim, C. Park, and J. Oh: A traffic control system to manage bandwidth usage in IP networks supporting Differentiated Service," Korea, (2001).
- [10] B. Gaidioz, P. Primet,: The Equivalent Differentiated Services Model, (2002).
- [11] X. Xiao, A. Hannan, and B. Bailey: Traffic Engineering with MPLS in the Internet," IEEE Network Magazine, vol. 14, (2000).
- [12] S. Manjanatha, R. Barto: Integrating Differentiated Services with ATM" published by Springer Netherlands, Volume 19, Numbers 3-4, pp 403-423(2002)
- [13] E. Park and C. Choi: Adaptive Token Bucket Algorithm for Fair Bandwidth Allocation in DiffServ Networks", Seoul National University, Seoul, KOREA ,Global Telecommunications Conference, Volume 6, Issue , PP. 3176 – 3180 (2003)
- [14] S.Sudha, N. Ammasaigounden :Adaptive Token Allocation Algorithm For Enhancing The Fairness Among Long Live TCP Flows In Diffserv, (2008)
- [15] S. Vegesna: IP Quality of Service, Cisco Press, (2001)
- [16] J. Cao, W. S. Cleveland, D. Lin, D. X. Sun: Internet Traffic Tends Toward Poisson and Independent as the Load Increases, Springer, New York, (2002).
- [17] M. Flannagan,: Administering Cisco QOS for IP Networks ,Syngress ( 2001).
- [18] C. Sanders: Practical Packet Analysis: Using Wireshark to Solve Real-World Network Problems, No Starch Press, (2007).