İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

QUALITY OF SERVICE ENHANCEMENT IN IP BASED NETWORKS USING DIFFSERV

M.Sc. Thesis by
Okan VURAL, Control and Comp. Eng.
(504001561)

Department : Computer Engineering
Programme: Computer Engineering

JANUARY 2004

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

QUALITY OF SERVICE ENHANCEMENT IN IP BASED NETWORKS USING DIFFSERV

M.Sc. Thesis by

Okan VURAL, Control and Comp. Eng.

(504001561)

Date of submission : 05 May 2003

Date of defence examination : 13 Jan 2004

Supervisor (Chairman) : Assoc. Prof. Dr. Sema OKTUĞ

Members of the Examining Committee Prof. Dr. Emre HARMANCI

Prof.Dr. Cem ERSOY

JANUARY 2004

PREFACE

I would like to thank my advisor, Assoc. Prof. Dr. Sema OKTUĞ for her all guidance, encouragement, suggestions and help throughout my entire graduate study.

I would also like to thank to Phd. student Bahri OKUROĞLU for his useful guidance, discussions and conversations.

I am grateful to my family for their care and encouragement.

May 2003 Okan VURAL

TABLE OF CONTENTS

ABBREVIATIONS

AC : Admission Control
AF : Assured Forwarding
AS : Assured Service
AS : Autonomous System
BA : Behavior Aggregate
BB : Bandwidth Broker
BE : Best Effort
CBQ : Class Based Queuing
CLS : Controlled-Load Service
CU : Currently Unused
DiffServ : Differentiated Services
DS : Differentiated Services
DSCP : Differentiated Services CodePoint
EF : Expedited Forwarding
EMBAC : End-to-End Measurement-Based Admission Control
FEC : Forward Error Correction
FIFO : First In First Out
GOP : Group of Pictures
GS : Guaranteed Service
IETF : Internet Engineering Task Force
IntServ : Integrated Services
IP : Internet Protocol
IPv4 : Internet Protocol Version 4
IPv6 : Internet Protocol Version 6
ITU : International Telecommunication Union
LDP : Label Distribution Protocol
MBAC : Measurement-Based Admission Control
MF : Multi Field
MPLS : Multi-Protocol Label Switching
PDB : Per-Domain Behavior
PHB : Per Hop Behaviour
PQ : Priority Queuing
PSTN : Public Switched Telephone Networks
QoS : Quality of Service
RATES : Routing And Traffic Engineering Server
RED : Random Early Drop
RSVP : ReSerVation Protocol
SLA : Service Level Agreement
SLS : Service Level Specification
SPQ : Strict Priority Queueing
TC : Traffic Class
TCA : Traffic Conditioning Agreement
TOS : Type of Service
VLL : Virtual Leased Line
VPN : Virtual Private Network
WFQ : Weighted Fair Queueing
WG : Working Group
WRR : Weighted Round Robin

TABLE LIST

FIGURE LIST

DIFFSERV MİMARİSİ İLE IP TABANLI AĞLARDA SERVİS KALİTESİ GELİŞTİRME

ÖZET

Bu çalışmada Ayrım Gözeten Servis (*DiffServ*) mimarisi ile IP tabanlı ağlarda servis kalitesi geliştirme tartışılmıştır. Son on-on beş yılda IP tabanlı ağlarda bir servis kalitesinden bahsetmek mümkün değildi. İnternet'te bugün de desteklenen tek servis türü Elden Gelenin En İyisi (*Best Effort*) adı verilen servis türüdür. Bu serviste, ağ, paketleri hedeflerine herhangi bir garanti ya da özel kaynak ayrımı yapmaksızın ulaştırır. Başka bir deyişle, paketler herhangi bir zaman ya da paket kaybı açısından garanti verilmeksizin, mümkün olan en hızlı ve en verimli şekilde yönlendirilirler. *Best-Effort* servisinde uç düğümler tıkanıklık ve hata denetimi bakımından daha karmaşık bir yapıdadır, böylece ağ düğümleri nispeten daha basit bir yapıdadırlar. Bu durum, İnternet'in büyümesine ölçeklenebilirlik bakımından bir fayda sağlamıştır. Fakat servis isteyen kullanıcı sayısı ağ kapasitesinin üstüne çıktıkça, yeni istekler reddedilmez bunun yerine servis alan mevcut kullanıcıların aldıkları servis kalitesi düşürülür.

İnternetin daha ticari bir boyuta ulaşması ve gelişen multimedya gibi uygulamaların ihtiyaç duydukları servis kalitesinin artmasıyla (İnternet telefon ve video uygulamaları gibi) İnternet'te bir servis kalitesi sağlanma ihtiyacı zorunlu bir konuma gelmiştir. Bu zorunluluk, farklı mimarilerin doğmasına yol açmıştır: Bütünleşik Servis (*IntServ*) ve Ayrım Gözeten Servis (*DiffServ*) olmak üzere.

Integrated Servis (IntServ) IETF tarafından servis kalitesi sağlamak üzere önerilmiş bir mimaridir. *Kaynak ayrımı* ile karakterize edilir. Uygulamaların ihtiyaç duydukları servis kalitesinin sağlanması için kaynakların önceden atanması varsayımına dayanır. *IntServ* mimarisi kendi içinde iki alt sınfa ayrılır: *Garantili Servis (Guaranteed Service)* ve *Kontrollü-Yük Servisi (Controlled-Load Service)*.

Integrated Servis ve *RSVP* (*Resource Reservation Protocol*) kaynak ayrımı protokolünün uygulamadaki zorlukları ve ölçeklenebilirlik problemi nedeniyle *DiffServ* adı verilen yeni bir servis kalitesi sağlayan mimari doğmuştur. DiffServ'ün amacı; her bir akışa garanti vermek yerine ölçeklenebilir farklı servis sınıfları yaratarak bir servis kalitesi sunmaktır. Ölçeklenebilirliğini ise *Per-Hop Behaviours* (PHB) adı verilen her bir servis sınıfına uygulanacak davranışları belirleyerek sağlar. DiffServ mimarisi IntServ mimarisi gibi iki alt sınıf tanımlar: *Garantili Servis (Guaranteed Service)* ve *İkincil Servis (Expedited Service)*.

Bu tez çalışmasında bazı DiffServ uygulamaları artı ve eksileriyle ele alınmıştır. Yaptığımız ana çalışma ise Band Genişliği Yöneticili bir DiffServ gerçeklemesidir. Bu çalışmada, üç servis sınıfı kuyruklarda RED algoritması parametreleriyle kontrol edilmektedir. Ağın bandgenişliğini yöneten bir yönetici vardır ve bu yönetici kuyruklardan belli periyodlarla aldığı iletim bilgileriyle ağın durumu hakkında bilgi alır. Böylece yeni istekleri kabul eder ya da red cevabı verir. Bu gerçeklemeler NS2 benzetim ortamında ve C++ programlama dili kullanılarak yapılmıştır. Sonuçlar ve tartışmalar verilmiştir.

QUALITY OF SERVICE ENHANCEMENT IN IP BASED NETWORKS USING DIFFSERV

SUMMARY

In this study, improving *Quality of Service (QoS)* on the Internet with DiffServ architecture is discussed. Within the past decade, it is certainly not supported for Quality of Service (QoS) over the IP-based Internet. The Internet as it stands today only support one service class called *Best-Effort (BE)* Service. In this service class the network would make an earnest attempt to deliver packets to their destinations but with no guarantees or special resources allocated for any of the packets. With another words, traffic is processed as quickly as possible but there is no guarantee as to timeliness (i.e., jitter) or actual delivery or even how much can be delivered (i.e., throughput). Best-Effort IP allows the complexity to stay in the end-hosts so the network can remain relatively simple. This scales well as evidenced by the Internet to support its phenomenal growth. As more hosts are connected, network service demands eventually exceeds its capacity, but service is not denied, instead it degrades gracefully. With the rapid transformation of the Internet into a commercial infrastructure, and the increasing needs of applications such as multimedia, IP telephony (two-way applications), demands for Quality of Service (QoS) have rapidly developed. This need was resulted to different architectures: *IntServ* and *DiffServ*.

Integrated Services (IntServ) is a QoS mechanism proposed by the *IETF* and is characterised by resource reservation. The assumption is that resources must be explicitly managed in order to meet application requirements. The IntServ architecture proposes two more classes of service: *Guaranteed Service* and *Controlled-Load Service*.

Due to the difficulties encountered in implementing and deploying the IntServ/RSVP architecture, another QoS mechanism known as *Differentiated Services* (DiffServ) was proposed. The goal of DiffServ is to give a scalable service discrimination without the need of per-flow state and signaling at every hop or router as in IntServ. The DiffServ architecture achieves its scaling properties by defining a small number of different packet forwarding treatments known as *Per-Hop Behaviours* (PHB). The DiffServ architecture also proposes two more classes of service like IntServ: *Assured Service* and *Expedited Service*.

In the study some DiffServ implementations are discussed with their problems and gains. Our main implementation is on a Bandwidth Broker based DiffServ implementation. In this study, BB plays as bandwidth manager role in the topology. In the topology, there are three service classes and all queues send their link state informations (how many packets of each class is sent or dropped) to the BB. And BB manages the QoS resources according to the needs of the sources. This implementation is made on NS2 simulation platform with C++ code. The results obtained and discussions are also given in the thesis.

1. INTRODUCTION

The Internet as it stands today only supports one service class called -*Best-Effort (BE) Service*. The network would make an earnest attempt to deliver packets to their destinations but with no guarantees and/or special resources allocated for any of the packets. In other words, traffic is processed as quickly as possible but there is no guarantee for timeliness or actual delivery or even how much can be delivered (i.e. throughput). With the rapid transformation of the Internet into a commercial infrastructure, demands for Quality of Service (QoS) have also developed [44].

By default, Internet Protocol (IP)-based networks provide Best-Effort data delivery. Best-Effort IP allows the complexity to stay in the end-hosts so the network can remain relatively simple [21]. As more hosts are connected, network service demands eventually exceeds its capacity, and service quality degrades gracefully. The service quality degradation results variability in delivery delays (i.e., jitter) and packet losses but these do not adversely affect typical Internet applications such as email, file transfer and web applications.

However, this is a problem particularly for applications with real-time requirements such as those that deliver multimedia, the most demanding of which are two-way applications like telephony and video conferencing [43]. Therefore to provide possibility for transporting these types of applications through the Internet, methods to reserve resources for their packets or to differentiate their packets (from *BE traffic*)-like giving privileges to those packets- are needed. The QoS need was resulted in different architectures: *IntServ* (*Integrated Service*) and *DiffServ* (*Differentiated Service*).

IntServ is a QoS mechanism proposed by the *IETF* and is characterised by resource reservation. The assumption is that resources must be explicitly managed in order to meet application requirements. The IntServ architecture proposes two more classes of service: *Guaranteed Service* and *Controlled-Load Service.*

Due to the problems encountered in implementing and deploying the IntServ/RSVP (*Resource Reservation Protocol*) architecture, another QoS mechanism known as DiffServ was proposed. The goal of DiffServ is to give a scalable service discrimination without the need of per-flow state and signaling at every hop or router as in IntServ. The DiffServ architecture achieves its scaling properties by defining a small number of different packet forwarding treatments known as *Per-Hop*

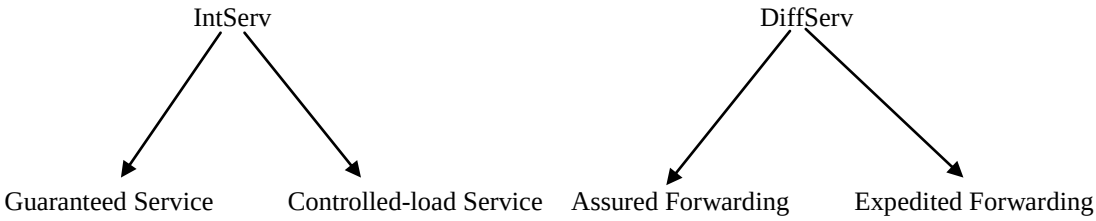
Behaviours (PHB). The DiffServ architecture also proposes two more classes of service like IntServ: *Assured Service* and *Expedited Service*.

Although, there is more than one way to characterise Quality of Service, in general, QoS is the ability of a network element (e.g., an application, a host, a router) to provide some level of assurance for consistent network data delivery services. This thesis deals with improving Quality of Service (QoS) on the Internet.

In this study, some DiffServ implementations are discussed with their problems and gains. Our main implementation is a Bandwidth Broker (*BB)* based DiffServ implementation. In this study, BB plays a bandwidth manager role in the topology. In the topology, there are three service classes and all queues send their link state information (how many packets of each class is sent or dropped) to the BB. The BB manages system resources according to the needs of the sources. The BB implementation is added to NS2 simulation platform using C++ modules. The results obtained and discussions are also presented in the thesis.

The thesis is organized as follows: in Chapter 2.1, *IntServ* architecure is detailed with RSVP and with its subservice classes. In Chapter 2.2, the *DiffServ* architecture is described much more detailed than *IntServ*. In Chapter 3 the admission control mechanisms in *DiffServ* architecture are given. The literature survey is given in the Chapter 4. The problems of real time applications in the Internet and some solutions described in the literature are given in Chapter 5. In Chapter 6 the detailed implementation with NS2 and simulation results are given. And the Section 7 concludes and gives reference to the future work.

2. QUALITY OF SERVICE ENHANCEMENT IN IP BASED NETWORKS



2.1 Integrated Services And RSVP

Integrated Services (IntServ) is a QoS mechanism proposed by the IETF [31] and is characterised by resource reservation [44]. The assumption is that resources (e.g., bandwidth) must be explicitly managed in order to meet application requirements. *Resource Reservation*, *Admission Control (AC)*[27], *Packet Scheduling* and *Buffer Management* [86] are the key building blocks of the service [27]. In addition to the traditional *Best-Effort service*, the IntServ architecture proposes two more classes of service; *Guaranteed Service* and *Controlled-Load Service*. All three classes (also BE Service) are briefly discussed below.

2.1.1 Guaranteed Service

This service is intended for delay-sensitive applications (such as multimedia applications). As the name suggests, service is guaranteed because there is an assured amount of bandwidth (and other resources) that has been reserved. It also assures firm end-to-end delay bounds with no packet loss when conforming to the parameters negotiated at the connection setup period [30]. The *GS* is characterized by peak rate, token bucket parameters and maximum packet size. Network utilization in GS is usually acceptable when flows are smooth. When flows are bursty, GS results in low utilization due to its worst-case service commitments [86].

2.1.2 Controlled-Load Service

This service is somehow between *Best-Effort* and *Guaranteed services*. It gives better quality than *Best-Effort* but cannot provide strictly bounded service the *Guaranteed service* promises. It is usually for applications requiring probabilistic delay bounds. Under lightly loaded network conditions, this service closely approximates to the Best-Effort service [43]. It uses loose admission control and

simple queue mechanisms, and is essentially for adaptive real-time communications. Thus, it does not provide a worst-case delay bound like the GS. CLS traffic is also characterized by peak rate, token bucket parameters, and maximum packet size. Some capacity (admission) control is needed to ensure that this service is received even when the node is overloaded [86].

One possible implemetation of CLS is to provide a queueing mechanism with two priority levels: a high priority for controlled-load and a lower priority for best-effort service. An admission control algorithm is used to limit the amount of traffic placed in the high-priority queue.

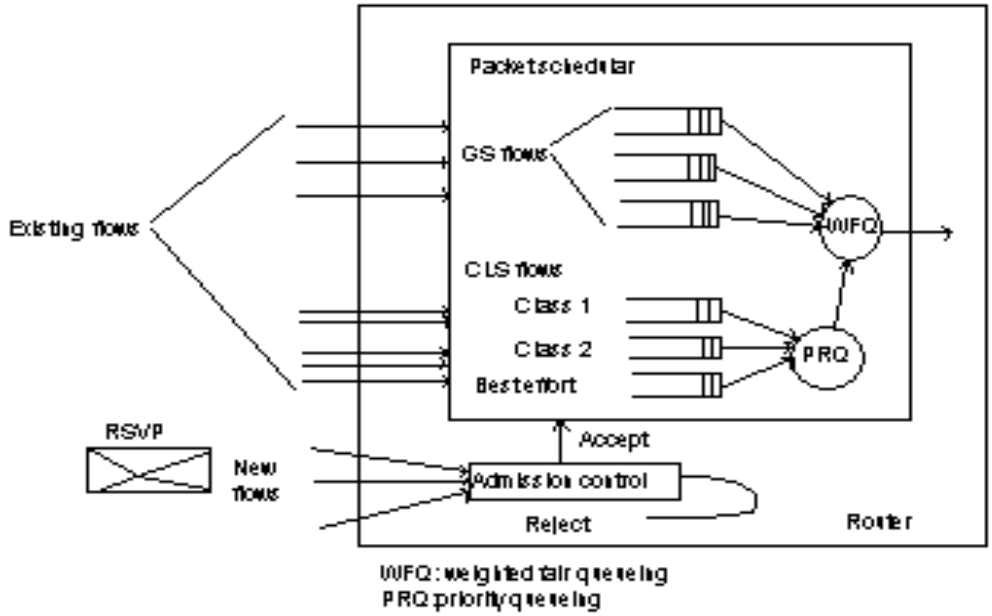


Figure 2.1.2 An IntServ implementation framework [86]

2.1.3 Best-Effort Service

In this service type, the network makes its best attempt to deliver the packets to their destinations but with no guarantees and no special resources allocated for any of the packets. The forwarding of packets is completely egalitarian; all packets receive the same quality of service, and packets are typically forwarded using a strict *FIFO* queuing discipline [43][8].

2.1.4 CAC in IntServ

2.1.4.1 RSVP

In order for the applications to communicate their QoS requirements to nodes along the transit path, a signaling mechanism is required. The *IETF IntServ Working Group (WG)* recommends RSVP (whose specification is given in [28]) as the signaling protocol to reserve resources. The main function of RSVP is to provide QoS request (on behalf of the application traffic) to all routers along the transit path and to maintain the state information in the routers for each data flow [29].

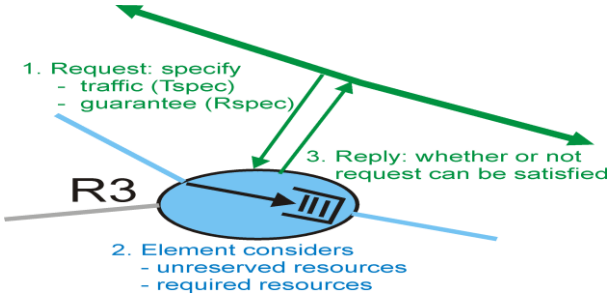


Figure 2.1.4.1 General concept of CAC and RSVP [45].

Figure 2.1.4.2 shows the signaling process used by RSVP to request for network resources.

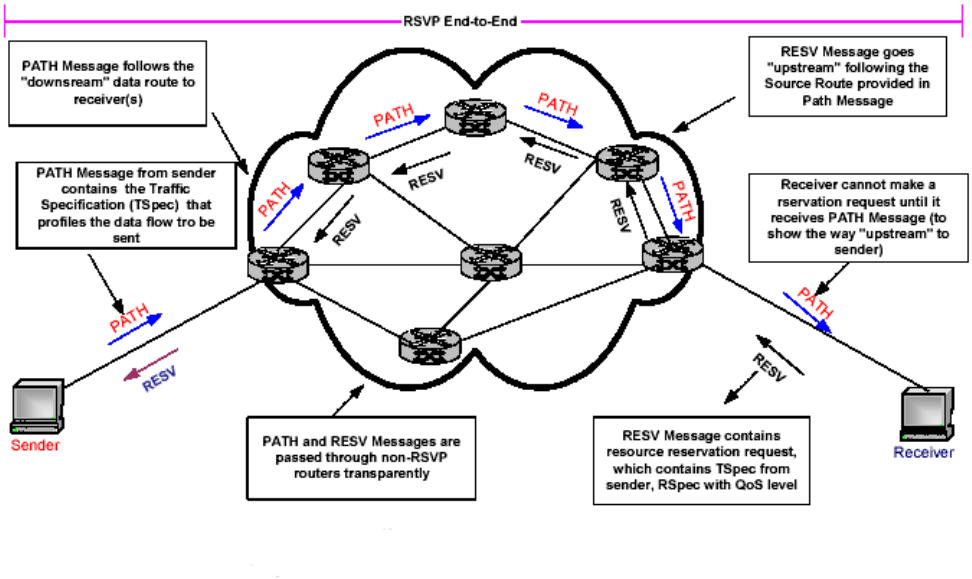


Figure 2.1.4.2 RSVP, *PATH* and *RESV* messages used to establish a resource reservation between a sender and a receiver [43].

Structure of the *PATH* message

Before data packets are sent over the link, a *PATH* message is send to the receiver. This particular message will specify a description of the traffic, which includes the identity of the sender, its application, traffic profile (bandwidth and the burst characteristics) and the classification criteria, which the traffic can be recognized. These classification criteria are the source and destination addresses, source and destination IP ports that will uniquely identify packets belonging to a particular flow [28].

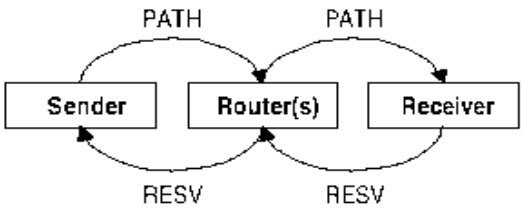


Figure 2.1.4.3 RSVP Mechanism [28].

RSVP Signaling

The signaling process works as follows: The sender sends a *PATH* message to the receiver specifying the characteristics of the traffic. Each intermediate RSVP-enabled router along the *PATH* (i.e., downstream route) establishes a *PATH*-state that includes the previous source address of the *PATH* message and then forwards the message to the next hop (determined by the routing protocol).

When the receiver gets a *PATH* message, it responds with a *RESV* (reservation request) message to request resources for the flow. The *RESV* message includes the QoS level required. When an *RSVP* router receives a *RESV* message, it uses admission control procedures to authenticate the request and allocates the necessary resources. If the request cannot be satisfied due to lack of resources, the router returns an error back to the receiver and the signaling process will terminate. If the request is accepted, link bandwidth and buffer space are allocated and the related flow state information will be installed in the router. The router then sends RSVP message to the next router upstream (i.e., towards sender). When the last router receives the RESV message and accepts the request, it sends a confirmation message to the receiver. The flow is then established. As stated in [43] in this respect, IntServ model provides the closest thing to circuit emulation in IP networks.

Advantage of *RSVP*

This signaling protocol is very strong in providing QoS support.

Disadvantages of *RSVP*

1. It is not scalable, since it is necessary to maintain a flow state in each router along the flow's path, and all routers participate in the signaling protocol.

2. The number of *RSVP* messages processed is proportional to the number of flows in the network and bandwidth must be reserved in each router on a per-flow basis.

3. Both of these disadvantages can lead to poor router performance [22][44].

2.1.4.2 Aggregation in IntServ

Aggregation [61] is a mechanism used to reduce the number of signaling messages in an IntServ architecture. In this technique the admission control is only performed on an aggregated set of flows and therefore core routers need only to maintain the reservation state of each aggregate not of the flows. The RSVP protocol is used but only for the aggregate. Thus, core routers do not store the reservation state of individual flows. More specifically, when a flow asks for admission, the ingress router performs the admission control decision based solely on its knowledge of the bandwidth occupancy of the aggregate. To allow for load fluctuations, the ingress router can adjust reservations in the core at slow time scales when compared to the IntServ reservation time scale. Thus, the signaling and the amount of stored state information in the core routers can be highly reduced.

Advantage and Disadvantages of Aggregation in IntServ

The aggregation implies a tradeoff: with more aggregation, more flows are not admitted and the utilization decreases; with small aggregation the decrease in utilization is neglected but the number of signaling messages remains high. If loads are relatively constant, the nodes rarely need to be signalled. Otherwise the signaling will be near to IntServ's one.

2.2 Differentiated Service

Due to the difficulties encountered in implementing and deploying the IntServ/RSVP architecture, another QoS mechanism known as *Differentiated Services* (DiffServ) was proposed in [33]. The goal of DiffServ is to give a scalable service discrimination without the need of per-flow state and signaling at every hop or router as in IntServ. The DiffServ architecture achieves its scaling properties by defining a small number of different packet forwarding treatments known as *Per-Hop Behaviours* (PHB) [40].

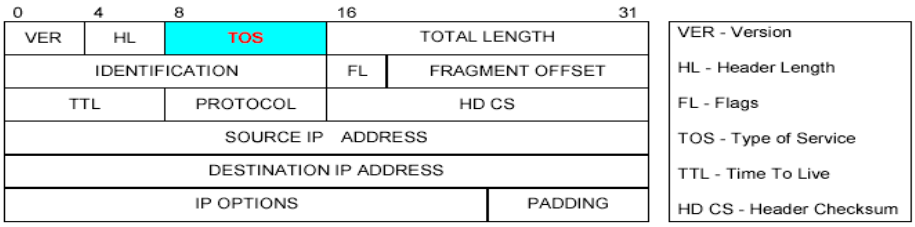


Figure 2.2.1 TOS byte in the IPv4 header [9]

DiffServ makes use of the *Type of Service* (TOS) byte (termed as DS field) in the IPv4 header (Figure 2.2.1) (which corresponds to the *Traffic Class, TC,* octet in the *IPv6* header (Figure 2.2.2 and Figure 2.2.3)). Each packet receives a particular forwarding treatment based on the marking in its IP TOS octet (now called *DS Code Point*). Packets marked same are treated the same way. There is no per-flow state required inside the network; core devices know only markings and not flows. Per-flow state is kept at the network edge and flows are aggregated based on desired behaviour. Figure 7 shows the components that work together to form a DiffServ domain. The *Bandwidth Broker* implements dynamic allocation of resources and is also responsible for making sure that network resources, both within the DiffServ domain and on links connecting adjacent domains, are properly provisioned.

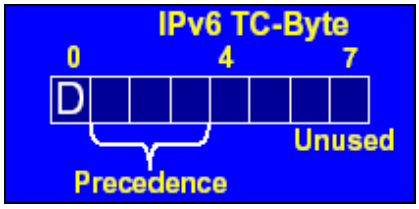


Figure 2.2.2 TC byte in the IPv6 header [85].

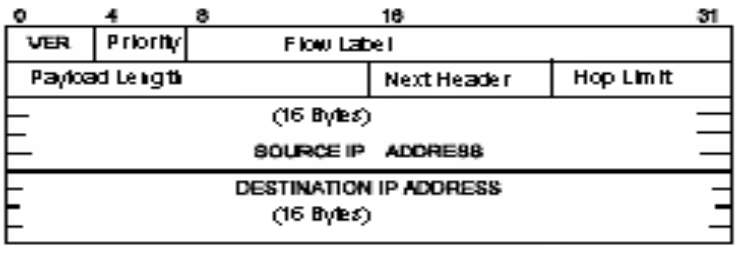


Figure 2.2.3 Ipv6 Fixed Header

By marking the DS fields of packets differently and handling packets based on their DS fields, several differentiated service classes can be created. The IETF DiffServ group has defined two classes of supporting applications the *Premium Service* & *the Assured Service* models.

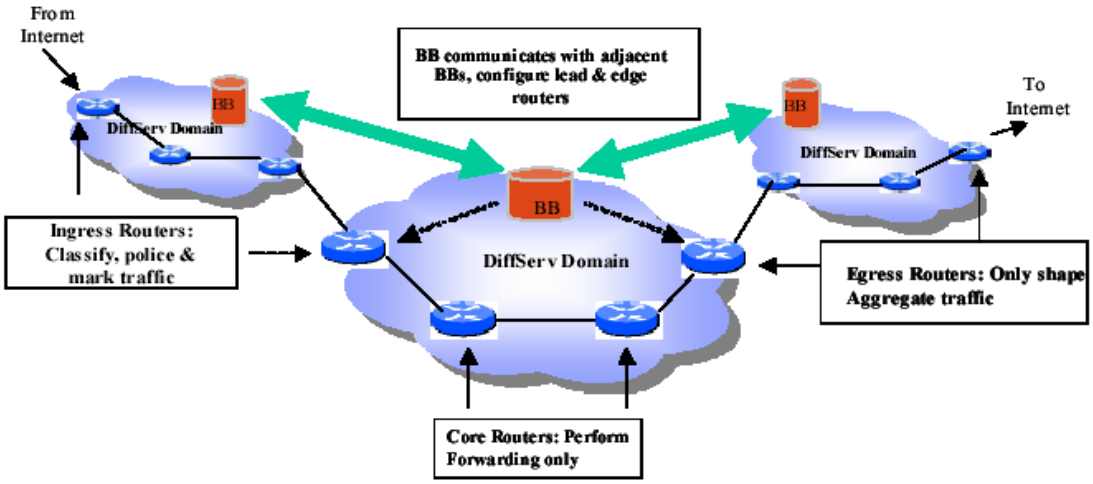


Figure 2.2.4 DiffServ Domain

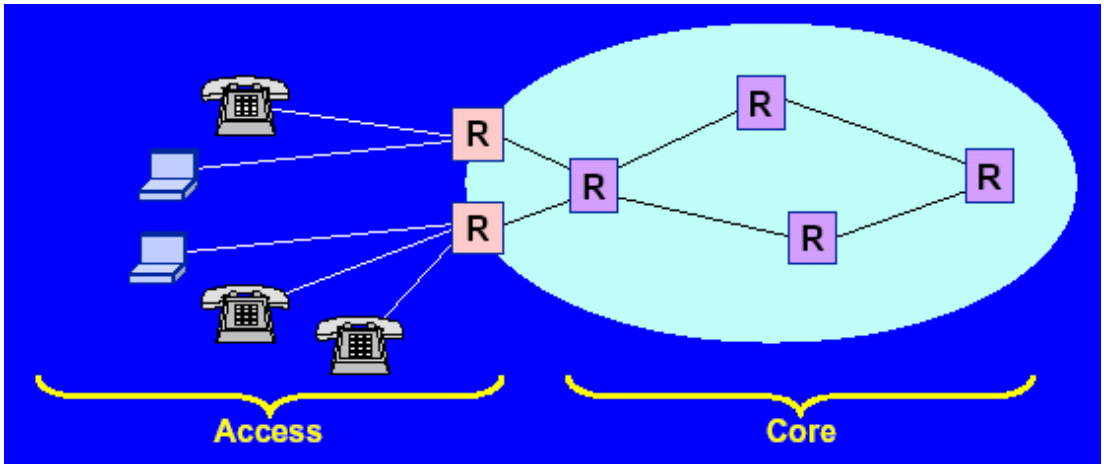


Figure 2.2.5. Different user types and access points to the DiffServ domain [85]

2.2.1 Service Classes of DiffServ

2.2.1.1 Assured Forwarding (AF)

The AF PHB group provides delivery of IP packets in four independently forwarded AF classes. Within each AF class, an IP packet can be assigned one of three different

levels of drop precedence. A DS node does not reorder IP packets of the same microflow if they belong to the same AF class.

There is a demand to provide assured forwarding of IP packets over the Internet. In a typical application, a company uses the Internet to interconnect its geographically distributed sites and wants an assurance that IP packets within this intranet are forwarded with high probability as long as the aggregate traffic from each site does not exceed the subscribed information rate (or profile). It is desirable that a site may exceed the subscribed profile with the understanding that the excess traffic is not delivered with as high probability as the traffic that is within the profile.

Assured Forwarding PHB group is a means for a provider DS domain to offer different levels of forwarding assurances for IP packets received from a customer DS domain. Four AF classes are defined, where each AF class in each DS node is allocated a certain amount of forwarding resources (buffer space and bandwidth e.g.). IP packets that wish to use the services provided by the AF PHB group are assigned by the customer or the provider DS domain into one or more of these AF classes according to the services that the customer has subscribed to.

Within each AF class IP packets are marked (again by the customer or the provider DS domain) with one of three possible drop precedence values. In case of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested DS node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value.

In a DS node, the level of forwarding assurance of an IP packet depends on;

(1) how much forwarding resources has been allocated to the AF class that the packet belongs to,

(2) what is the current load of the AF class, and, in case of congestion within the class,

(3) what is the drop precedence of the packet.

For example, if traffic conditioning actions at the ingress of the provider DS domain make sure that an AF class in the DS nodes is only moderately loaded by packets with the lowest drop precedence value and is not overloaded by packets with the two lowest drop precedence values, then the AF class can offer a high level of forwarding assurance for packets that are within the subscribed profile (i.e., marked with the

lowest drop precedence value) and offer up to two lower levels of forwarding assurance for the excess traffic.

2.2.1.2 The AF PHB Group

Assured Forwarding (AF) PHB group provides forwarding of IP packets in *N* independent AF classes. Within each AF class, an IP packet is assigned one of *M* different levels of drop precedence. An IP packet that belongs to an AF class *i* and has drop precedence *j* is marked with the AF codepoint AF_{ij}, where 1 <= *i*<= *N* and 1 <= *j* <= *M*. Currently, four classes (*N=4*) with three levels of drop precedence in each class (*M=3*) are defined for general use. More AF classes or levels of drop precedence may be defined for local use. A DS node should implement all four general use AF classes. Packets in one AF class must be forwarded independently from packets in another AF class, i.e., a DS node must not aggregate two or more AF classes together.

A DS node must allocate a configurable, minimum amount of forwarding resources to each implemented AF class. Each class should be serviced in a manner to achieve the configured service rate (bandwidth) over both small and large time scales.[72]

2.2.1.3 Queueing and discard behavior

An AF implementation must attempt to minimize long-term congestion within each class, while allowing short-term congestion resulting from bursts. This requires an active queue management algorithm. An example of such an algorithm is *Random Early Drop (RED)* [73].

2.2.1.4 Recommended codepoints for AF

Recommended codepoints for the four general use AF classes are given in the Table 2.2.1.4.

Table 2.2.1.4. Recommended AF codepoint values.

	Class 1	**Class 2**	**Class 3**	**Class 4**
Low Drop Prec.	001010	010010	011010	100010
Medium Drop Prec.	001100	010100	011100	100100
High Drop Prec.	001110	010110	011110	100110

2.2.1.5 Example Services

The AF PHB group could be used to implement, for example, the so-called *Olympic service*, which consists of three service classes**:** *bronze, silver,* and *gold*. Packets are assigned to these three classes so that packets in the gold class experience lighter load (and thus have greater probability for timely forwarding) than packets assigned to the silver class. Same kind of relationship exists between the silver class and the bronze class. If desired, packets within each class may be further separated by giving them either low, medium, or high drop precedence.

The bronze, silver, and gold service classes could in the network be mapped to the AF classes 1, 2, and 3. Similarly, low, medium, and high drop precedence may be mapped to AF drop precedence levels 1, 2, or 3.

The drop precedence level of a packet could be assigned, for example, by using a leaky bucket traffic policer, which has as its parameters a rate and a size, which is the sum of two burst values: a *committed burst size* and an *excess burst size*. A packet is assigned low drop precedence if the number of tokens in the bucket is greater than the excess burst size, medium drop precedence if the number of tokens in the bucket is greater than zero, but at most the excess burst size, and high drop precedence if the bucket is empty. It may also be necessary to set an upper limit to the amount of high drop precedence traffic from a customer DS domain in order to avoid the situation where a big amount of undeliverable high drop precedence packets from one customer DS domain can deny service to possibly deliverable high drop precedence packets from other domains.

Another way to assign the drop precedence level of a packet could be to limit the user traffic of an *Olympic service* class to a given peak rate and distribute it evenly across each level of drop precedence. This would yield a proportional bandwidth service, which equally apportions available capacity during times of congestion under the assumption that customers with high bandwidth microflows have subscribed to higher peak rates than customers with low bandwidth microflows.

The AF PHB group could also be used to implement a loss and low latency service using an over provisioned AF class, if the maximum arrival rate to that class is known a priori in each DS node. If low loss is not an objective, a low latency service could be implemented without over provisioning by setting a low maximum limit to the buffer space available for an AF class [72].

2.2.2 An expedited forwarding (EF) PHB

Network nodes that implement the differentiated services enhancements to IP use a codepoint in the IP header to select a per-hop behavior (PHB) as the specific forwarding treatment for that packet [47, 33]. The EF PHB can be used to build a low loss, low latency, low jitter, assured bandwidth, end-to-end service through DS domains. Such a service appears to the endpoints like a point-to-point connection or a *virtual leased line*. This service has also been described as *Premium service*.

Loss, latency and jitter are all due to the queues traffic experiences while transiting the network. Therefore providing low loss, latency and jitter for some traffic aggregate means ensuring that the aggregate sees no (or very small) queues. Queues arise when (short-term) traffic arrival rate exceeds departure rate at some node. Thus a service that ensures no queues for some aggregate is equivalent to bounding rates such that, at every transit node, the aggregate's maximum arrival rate is less than that aggregate's minimum departure rate.

Creating such a service has two parts:

1) Configuring nodes so that the aggregate has a well-defined minimum departure rate. ("Well-defined" means independent of the dynamic state of the node. In particular, independent of the intensity of other traffic at the node.)

2) Conditioning the aggregate (via policing and shaping) so that its arrival rate at any node is always less than that node's configured minimum departure rate.

The EF PHB provides the first part of the service. The network boundary traffic conditioners described in [33] provide the second part.

The EF PHB is not a mandatory part of the Differentiated Services architecture, i.e., a node is not required to implement the EF PHB in order to be considered DS-compliant. However, when a DS-compliant node claims to implement the EF PHB, the implementation must conform to the specification given in this document.

2.2.2.1 Description of EF per-hop behavior

The EF PHB is defined as a forwarding treatment for a particular diffserv aggregate where the departure rate of the aggregate's packets from any diffserv node must equal or exceed a configurable rate. The EF traffic should receive this rate independent of the intensity of any other traffic attempting to transit the node. It should average at least the configured rate when measured over any time interval equal to or longer

than the time it takes to send an output link MTU sized packet at the configured rate. (Behavior at time scales shorter than a packet time at the configured rate is deliberately not specified.) The configured minimum rate must be settable by a network administrator (using whatever mechanism the node supports for non-volatile configuration).

If the EF PHB is implemented by a mechanism that allows unlimited preemption of other traffic (e.g., a priority queue), the implementation must include some means to limit the damage EF traffic could inflict on other traffic (e.g., a token bucket rate limiter).

Traffic that exceeds this limit must be discarded. This maximum EF rate, and burst size if appropriate (must be settable by a network administrator). The minimum and maximum rates may be the same and configured by a single parameter.

2.2.2.3 Example mechanisms to implement the EF PHB

Several types of queue scheduling mechanisms may be employed to deliver the forwarding behavior described and thus implement the EF PHB. A simple priority queue will give the appropriate behavior as long as there is no higher priority queue that could preempt the EF for more than a packet time at the configured rate. (This could be accomplished by having a rate policer such as a token bucket associated with each priority queue to bound how much the queue can starve other traffic.) [48].

It is also possible to use a single queue in a group of queues serviced by a weighted round robin scheduler where the share of the output bandwidth assigned to the EF queue is equal to the configured rate. This could be implemented, for example, using one PHB of a Class Selector Compliant set of PHBs [47].

Another possible implementation is a CBQ scheduler that gives the EF queue priority up to the configured rate. All of these mechanisms have the basic properties required for the EF PHB though different choices result in different ancillary behavior such as jitter seen by individual microflows.

2.2.2.4 Recommended codepoint for this PHB and mutability

Codepoint ‘101110’ is recommended for the EF PHB. Packets marked for EF PHB may be remarked at a DS domain boundary only to other codepoints that satisfy the EF PHB. Packets marked for EF PHBs should not be demoted or promoted to another PHB by a DS domain.

2.2.3 Differentiated services field

Differentiated services are intended to provide a framework to enable deployment of scalable service discrimination in the Internet. The differentiated services approach aims to speed deployment by separating the architecture into two major components;

1. *Packet forwarding* is the relatively simple task that needs to be performed on a per-packet basis as quickly as possible. Forwarding uses the packet header to find an entry in a routing table that determines the packet's output interface.

2. *Routing* sets the entries in the routing table and may need to reflect a range of transit and other policies as well as to keep track of route failures. Routing tables are maintained as a background process to the forwarding task Further, routing is the more complex task and it has continued to evolve over the past years.

2.2.3.1 Packet forwarding

The forwarding path behaviors include the differential treatment an individual packet receives, as implemented by queue service disciplines and/or queue management disciplines. These per-hop behaviors are required in network nodes to deliver differentiated treatment of packets no matter how end-to-end or intra-domain services are constructed. Main focus is on the general semantics of the behaviors rather than the specific mechanisms used to implement them since these behaviors will evolve less rapidly than the mechanisms.

Per-hop behaviors and mechanisms to select them on a per-packet basis can be deployed in network nodes today and it is the aspect of the differentiated services architecture that is being addressed first. In addition, the forwarding path may require that some monitoring, policing, and shaping be done on the network traffic designated for special treatment in order to enforce requirements associated with the delivery of the special treatment. The wide deployment of such traffic conditioners is also important to enable the construction of services, though their actual use in constructing services may evolve over time.

In the packet forwarding path, differentiated services are realized by mapping the codepoint contained in a field in the IP packet header to a particular forwarding treatment, at each network node along its path. The codepoints may be chosen from a set of mandatory values or may have purely local meaning. PHBs are expected to be implemented by employing a range of queue service and/or queue management disciplines on a network node's output interface queue: for example *weighted round-robin (WRR) queue servicing* or *drop-preference queue management (DPQM)*.

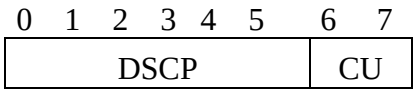
Marking is performed by traffic conditioners which are located at network boundaries, including the edges of the network (first-hop router or source host) and also by the administrative boundaries. *Traffic conditioners* may include the primitives of marking, metering, policing and shaping. Services are realized by the use of particular packet classification and traffic conditioning mechanisms at boundaries coupled with the concatenation of per-hop behaviors along the transit path of the traffic. As stated in [46] "*A goal of the differentiated services architecture is to specify these building blocks for future extensibility, both of the number and type of the building blocks and of the services built from them.*"

2.2.4 Differentiated services field definition

A replacement header field, called the *DS field*, is defined. This is intended to replace the existing definitions of the IPv4 TOS octet and the IPv6 Traffic Class octet.

Six bits of the DS field are used as a *codepoint (DSCP)* to select the PHB a packet experiences at each node. A two-bit *currently unused (CU)* field is reserved. The value of the CU bits are ignored by differentiated services-compliant nodes when determining the per-hop behavior to apply to a received packet.

The DS field structure is presented below:



DS-compliant nodes must select PHBs by matching against the entire 6-bit DSCP field, e.g., by treating the value of the field as a table index which is used to select a particular packet handling mechanism which has been implemented in that device. The value of the CU field must be ignored by PHB selection. The DSCP field is defined as an unstructured field to facilitate the definition of future per-hop behaviors.

Operators may choose to use different codepoints for a PHB, either in addition to or in place of the recommended default. Note that if operators do so choose, re-marking of DS fields may be necessary at administrative boundaries even if the same PHBs are implemented on both sides of the boundary.

2.2.5 IP Precedence history and evolution in brief

The IP Precedence field is something of a forerunner of the DS field. IP Precedence, and the IP Precedence Field, were first defined in [46]. The values that the three-bit

IP Precedence Field might take were assigned to various uses, including network control traffic, routing traffic, and various levels of privilege. The least level of privilege was deemed "routine traffic". In [46], the notion of Precedence was defined broadly as "An independent measure of the importance of this datagram." Not all values of the IP Precedence field were assumed to have meaning across boundaries, for instance "The Network Control precedence designation is intended to be used within a network only. The actual use and control of that designation is up to each network."

2.2.6 Example mechanisms for implementing class selector compliant

PHB Groups Class Selector Compliant PHBs can be realized by a variety of mechanisms, including *strict priority queueing (SPQ), weighted fair queueing (WFQ), WRR,* or variants *(RPS, HPFQA, DRR),* or *Class-Based Queuing (CBQ).*

It is important to note that these mechanisms might be available through other PHBs (standardized or not) that are available in a particular vendor's equipment. A network administrator might configure those routers to select the Strict Priority Queueing PHBs with codepoints 'xxx000'.

As another example given in [47], another vendor might employ a CBQ mechanism in its routers. The CBQ mechanism could be used to implement the Strict Priority Queueing PHBs as well as a set of Class Selector Compliant PHBs with a wider range of features than would be available in a set of PHBs that did no more than meet the minimum Class Selector PHB requirements.

2.2.7 Terminologies about DiffServ

Here are some terminologies about Diffserv that are given in the most Diffserv related RFCs:

- **Behavior Aggregate:** a collection of packets with the same codepoint crossing a link in a particular direction. The terms "*aggregate*" and "*behavior aggregate*" are used interchangeably in the RFCs.

- **Classifier:** an entity which selects packets based on the content of packet headers according to the predefined rules.

- **Class Selector Codepoint:** any of the eight codepoints in the range 'xxx000' (where 'x' may equal '0' or '1').

- **Class Selector Compliant PHB:** a per-hop behavior satisfying the Class Selector PHB.

- **Codepoint:** a specific value of the DSCP portion of the DS field. Recommended codepoints should map to specific, standardized PHBs. Multiple codepoints may map to the same PHB.

- **Differentiated Services Boundary:** the edge of a DS domain, where classifiers and traffic conditioners are likely to be deployed. A differentiated services boundary can be further sub-divided into ingress and egress nodes, where the **ingress/egress nodes** are the downstream/upstream nodes of a boundary link in a given traffic direction. A differentiated services boundary typically is found at the ingress to the first-hop differentiated services-compliant router (or network node) that a host's packets traverse, or at the egress of the last-hop differentiated services-compliant router or network node that packets traverse before arriving at a host. This is sometimes referred to as the boundary at a leaf router. A differentiated services boundary may be co-located with a host, subject to local policy.

- **Differentiated Services Domain:** a contiguous portion of the Internet over which a consistent set of differentiated services policies are administered in a coordinated fashion. A differentiated services domain can represent different administrative domains or autonomous systems, different trust regions, different network technologies (e.g., cell/frame), hosts and routers, etc.

- **Differentiated Services Field:** the IPv4 header TOS octet or the IPv6 Traffic Class octet when interpreted in conformance with the definitions given above.

- **Mechanism:** The implementation of one or more per-hop behaviors according to a particular algorithm.

- **Microflow:** a single instance of an application-to-application flow of packets which is identified by source address, destination address, protocol id, and source port, destination port (where applicable (e.g. in IntServ)).

- **Per-hop Behavior (PHB):** a description of the externally observable forwarding treatment applied at a differentiated services-compliant node to a behavior aggregate.

- **Per-hop Behavior Group:** a set of one or more PHBs that can only be meaningfully specified and implemented simultaneously, due to a common

constraint applying to all PHBs in the set such as a queue servicing or queue management policy.

- **Traffic Conditioning:** control functions that can be applied to a behavior aggregate, application flow, or other operationally useful subset of traffic, e.g., routing updates. These may include metering, policing, shaping, and packet marking. Traffic conditioning is used to enforce agreements between domains and to condition traffic to receive a differentiated service within a domain by marking packets with the appropriate codepoint in the DS field and by monitoring and altering the temporal characteristics of the aggregate where necessary.

- **Traffic Conditioner:** an entity that performs traffic conditioning functions and which may contain meters, policers, shapers, and markers. Traffic conditioners are typically deployed in DS boundary nodes (i.e., not in interior nodes of a DS domain).

- **Service:** a description of the overall treatment of (a subset of) a customer's traffic across a particular domain, across a set of interconnected DS domains, or end-to-end. Service descriptions are covered by administrative policy and services are constructed by applying traffic conditioning to create behavior aggregates which experience a known PHB at each node within the DS domain. Multiple services can be supported by a single per-hop behavior used in concert with a range of traffic conditioners.

To summarize, classifiers and traffic conditioners are used to select which packets are to be added to behavior aggregates. Aggregates receive differentiated treatment in a DS domain and traffic conditioners may alter the temporal characteristics of the aggregate to conform to some requirements. A packet's DS field is used to designate the packet's behavior aggregate and is subsequently used to determine which forwarding treatment the packet receives. A behavior aggregate classifier which can select a PHB, for example a differential output queue servicing discipline, based on the codepoint in the DS field should be included in all network nodes in a DS domain. The classifiers and traffic conditioners at DS boundaries are configured in accordance with some service specifications.

2.2.8 New terminologies and clarifications for Diffserv

As the Diffserv work has evolved, there have been several cases where terminology has needed to be created or the definitions in Diffserv standards track RFCs have

needed to be refined. Some minor technical clarifications were also found to be needed.

2.2.9 Terminology related to Service Level Agreements (SLAs)

The Diffserv Architecture [33] uses the term *Service Level Agreement* (SLA) to describe the "service contract... that specifies the forwarding service a customer should receive". The SLA may include traffic conditioning rules which (at least in part) constitute a Traffic Conditioning Agreement (TCA). A TCA is "an agreement specifying classifier rules and any corresponding traffic profiles and metering, marking, discarding and/or shaping rules which are to apply...."

As work progressed in Diffserv (as well as in the Policy WG [48]), it came to be believed that the notion of an "agreement" implied considerations that were of a pricing, contractual or other business nature, as well as those that were strictly technical. There also could be other technical considerations in such an agreement (e.g., service availability) which are not addressed by Diffserv. It was therefore agreed that the notions of SLAs and TCAs would be taken to represent the broader context, and that new terminology would be used to describe those elements of service and traffic conditioning that are addressed by Diffserv.

- **A Service Level Specification (SLS)** is a set of parameters and their values which together define the service offered to a traffic stream by a DS domain.

- **A Traffic Conditioning Specification (TCS)** is a set of parameters and their values which together specify a set of classifier rules and a traffic profile. A TCS is an integral element of an SLS.

- Note that the definition of *Traffic stream* is unchanged from [47]. A traffic stream can be an individual microflow or a group of microflows (i.e., in a source or destination DS domain) or it can be a BA. Thus, an SLS may apply in the source or destination DS domain to a single microflow or group of microflows, as well as to a BA in any DS domain.

- Also note that the definition of a *Service Provisioning Policy* is unchanged from [33] which defines it as "a policy which defines how traffic conditioners are configured on DS boundary nodes and how traffic streams are mapped to DS behavior aggregates to achieve a range of services." According to one definition given in [77], [48], a policy is "...a set of rules to administer, manage, and control access to network resources". Therefore, the relationship between an SLS and a

service provisioning policy is that the latter is, in part, the set of rules that express the parameters and range of values that may be in the former. Further note that this definition is more restrictive than that in [77].

2.2.10 Differentiated services architectural model

The differentiated services architecture is based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network, and assigned to different behavior aggregates. Each behavior aggregate is identified by a single DS codepoint. Within the core of the network, packets are forwarded according to the per-hop behavior associated with the DS codepoint.

2.2.10.1 Differentiated Services domain

A DS domain is a contiguous set of DS nodes which operate with a common service provisioning policy and set of PHB groups implemented on each node. A DS domain has a well-defined boundary consisting of DS boundary nodes which classify and possibly condition ingress traffic to ensure that packets which transit the domain are appropriately marked to select a PHB from one of the PHB groups supported within the domain. Nodes within the DS domain select the forwarding behavior for packets based on their DS codepoint, mapping that value to one of the supported PHBs using either the recommended codepoint->PHB mapping or a locally customized mapping. Inclusion of non-DS-compliant nodes within a DS domain may result in unpredictable performance and may impede the ability to satisfy service level agreements (SLAs).

A DS domain normally consists of one or more networks under the same administration; for example, an organization's intranet or an ISP. The administration of the domain is responsible for ensuring that adequate resources are provisioned and/or reserved to support the SLAs offered by the domain.

2.2.10.2 DS boundary nodes and interior nodes

A DS domain consists of DS boundary nodes and DS interior nodes. *DS boundary nodes* interconnect the DS domain to other DS or non-DS-capable domains, whilst *DS interior nodes* only connect to other DS interior or boundary nodes within the same DS domain.

Both DS boundary nodes and interior nodes must be able to apply the appropriate PHB to packets based on the DS codepoint; otherwise unpredictable behavior may result. In addition, DS boundary nodes may be required to perform traffic

conditioning functions as defined by a *traffic conditioning agreement (TCA)* between their DS domain and the peering domain which they connect to.

Interior nodes may be able to perform limited traffic conditioning functions such as DS codepoint re-marking. Interior nodes which implement more complex classification and traffic conditioning functions are analogous to DS boundary nodes.

A host in a network containing a DS domain may act as a DS boundary node for traffic from applications running on that host. If a host does not act as a boundary node, then the DS node topologically closest to that host acts as the DS boundary node for that host's traffic.

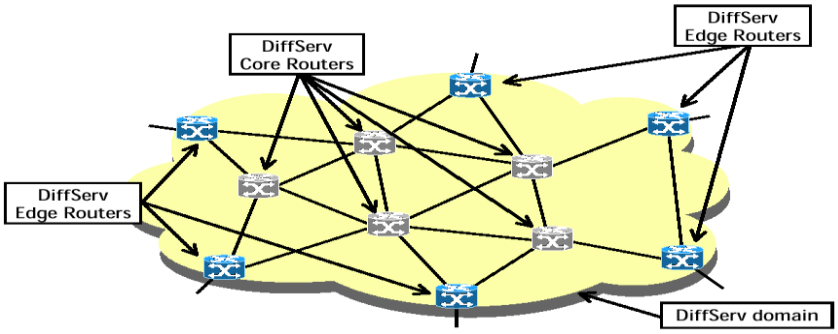


Figure 2.2.10.2.1 Differentiated Services: basic concept

2.2.10.3 DS ingress node and egress node

DS boundary nodes act both as a DS ingress node and as a DS egress node for different directions of traffic. Traffic enters a DS domain at a DS ingress node and leaves a DS domain at a DS egress node. A DS ingress node is responsible for ensuring that the traffic entering the DS domain conforms to any TCA between it and the other domain to which the ingress node is connected.

A DS egress node may perform traffic conditioning functions on traffic forwarded to a directly connected peering domain, depending on the details of the TCA between the two domains. Note that a DS boundary node may act as a DS interior node for some set of interfaces.

2.2.10.4 Differentiated Services region

A differentiated services region (DS Region) is a set of one or more contiguous DS domains. DS regions are capable of supporting differentiated services along paths which span the domains within the region.

The DS domains in a DS region may support different PHB groups internally and different codepoint->PHB mappings. However, to permit services which span across the domains, the peering DS domains must each establish a peering SLA which defines (either explicitly or implicitly) a TCA which specifies how transit traffic from one DS domain to another is conditioned at the boundary between the two DS domains.

It is possible that several DS domains within a DS region may adopt a common service provisioning policy and may support a common set of PHB groups and codepoint mappings, thus eliminating the need for traffic conditioning between those DS domains.

2.2.11 Traffic classification and conditioning

Differentiated services are extended across a DS domain boundary by establishing a *SLA* between an upstream network and a downstream DS domain. The SLA may specify packet classification and re-marking rules and may also specify traffic profiles and actions to traffic streams which are in- or out-of-profile. The *TCA* between the domains is derived (explicitly or implicitly) from this SLA.

The packet classification policy identifies the subset of traffic which may receive a differentiated service by being conditioned and/ or mapped to one or more behavior aggregates (by DS codepoint re-marking) within the DS domain.

Traffic conditioning performs metering, shaping, policing and/or re- marking to ensure that the traffic entering the DS domain conforms to the rules specified in the TCA, in accordance with the domain's service provisioning policy. The extent of traffic conditioning required is dependent on the specifics of the service offering, and may range from simple codepoint re-marking to complex policing and shaping operations.

2.2.11.1 Classifiers

Packet classifiers select packets in a traffic stream based on the content of some portion of the packet header. Two types of classifiers are:

1. **The BA (Behavior Aggregate) Classifier** classifies packets based on the DS codepoint only.

2. **The MF (Multi-Field) Classifier** selects packets based on the value of a combination of one or more header fields, such as source address, destination

address, DS field, protocol ID, source port and destination port numbers, and other information such as incoming interface.

Classifiers are used to "steer" packets matching some specified rule to an element of a traffic conditioner for further processing. Classifiers must be configured by some management procedure in accordance with the appropriate TCA. The classifier should authenticate the information which it uses to classify the packet.

2.2.11.2 Traffic profiles

A traffic profile specifies the temporal properties of a traffic stream selected by a classifier. It provides rules for determining whether a particular packet is in-profile or out-of-profile. For example, a profile based on a token bucket may look like:

codepoint = X, use token-bucket *r*, *b*

The above profile indicates that all packets marked with DS codepoint *X* should be measured against a token bucket meter with rate ***r*** and burst size ***b***. In this example out-of-profile packets are those packets in the traffic stream which arrive when insufficient tokens are available in the bucket. The concept of in- and out-of-profile can be extended to more than two levels, e.g., multiple levels of conformance with a profile may be defined and enforced.

Different conditioning actions may be applied to the in-profile packets and out-of-profile packets, or different accounting actions may be triggered. In-profile packets may be allowed to enter the DS domain without further conditioning; or, alternatively, their DS codepoint may be changed. The latter happens when the DS codepoint is set to a non-Default value for the first time, or when the packets enter a DS domain that uses a different PHB group or codepoint->PHB mapping policy for this traffic stream. Out-of- profile packets may be queued until they are in-profile (shaped), discarded (policed), marked with a new codepoint (re-marked), or forwarded unchanged while triggering some accounting procedure. Out-of-profile packets may be mapped to one or more behavior aggregates that are "inferior" in some dimension of forwarding performance to the BA into which in-profile packets are mapped.

Note that a traffic profile is an optional component of a TCA and its use is dependent on the specifics of the service offering and the domain's service provisioning policy.

2.2.11.3 Traffic conditioners

A traffic conditioner may contain the following elements: *meter, marker, shaper,* and *dropper*. A traffic stream is selected by a classifier, which steers the packets to a logical instance of a traffic conditioner. A meter is used (where appropriate) to measure the traffic stream against a traffic profile. The state of the meter with respect to a particular packet (e.g., whether it is in- or out-of-profile) may be used to affect a marking, dropping, or shaping action.

When packets exit the traffic conditioner of a DS boundary node the DS codepoint of each packet must be set to an appropriate value.

Fig. 2.2.11.3.1 shows the block diagram of a classifier and traffic conditioner. Note that a traffic conditioner may not necessarily contain all four elements. For example, in the case where no traffic profile is in effect, packets may only pass through a classifier and a marker.

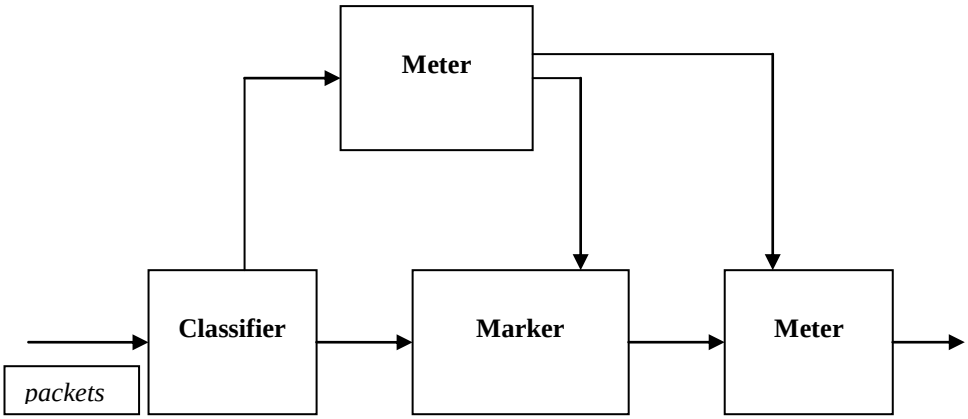


Fig 2.2.11.3.1 Logical View of a Packet Classifier and Traffic Conditioner

Meters

Traffic meters measure the temporal properties of the stream of packets selected by a classifier against a traffic profile specified in a TCA. 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 (to some extent).

Markers

Packet markers set the DS field of a packet to a particular codepoint, adding the marked packet to a particular DS behavior aggregate. The marker may be configured to mark all packets which are steered to it to a single codepoint, or may be configured to mark a packet to one of a set of codepoints used to select a PHB in a PHB group, according to the state of a meter. When the marker changes the codepoint in a packet it is said to have "*re-marked*" the packet.

Shapers

Shapers delay 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.

Droppers

Droppers discard 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 know 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.

2.2.11.4 Location of traffic conditioners and MF classifiers

Traffic conditioners are usually located within DS ingress and egress boundary nodes, but may also be located in nodes within the interior of a DS domain, or within a non-DS-capable domain.

2.2.12 Per-Hop Behaviors

A per-hop behavior (PHB) is a description of the externally observable forwarding behavior of a DS node applied to a particular DS behavior aggregate. *Forwarding behavior* is a general concept in this context. For example, in the event that only one behavior aggregate occupies a link, the observable forwarding behavior (i.e., loss, delay, jitter) will often depend only on the relative loading of the link (i.e., in the event that the behavior assumes a work-conserving scheduling discipline). Useful behavioral distinctions are mainly observed when multiple behavior aggregates compete for buffer and bandwidth resources on a node. The PHB is the means by which a node allocates resources to behavior aggregates, and it is on top of this basic hop-by-hop resource allocation mechanism that useful differentiated services may be constructed.

The most simple example of a PHB is one which guarantees a minimal bandwidth allocation of *X*% of a link (over some reasonable time interval) to a behavior aggregate. This PHB can be fairly easily measured under a variety of competing traffic conditions. A slightly more complex PHB would guarantee a minimal bandwidth allocation of *X%* of a link, with proportional fair sharing of any excess link capacity. In general, the observable behavior of a PHB may depend on certain constraints on the traffic characteristics of the associated behavior aggregate, or the characteristics of other behavior aggregates.

PHBs may be specified in terms of their resource (e.g., buffer, bandwidth) priority relative to other PHBs, or in terms of their relative observable traffic characteristics (e.g., delay, loss). These PHBs may be used as building blocks to allocate resources and should be specified as a group (PHB group) for consistency. PHB groups will usually share a common constraint applying to each PHB within the group, such as a packet scheduling or buffer management policy. The relationship between PHBs in a group may be in terms of absolute or relative priority (e.g., discard priority by means of deterministic or stochastic thresholds), but this is not required (e.g., *N* equal link shares). A single PHB defined in isolation is a special case of a PHB group.

2.2.12.1 Implementation of PHBs on nodes

PHBs are implemented in nodes by means of some buffer management and packet scheduling mechanisms. PHBs are defined in terms of behavior characteristics relevant to service provisioning policies, and not in terms of particular implementation mechanisms. In general, a variety of implementation mechanisms may be suitable for implementing a particular PHB group. Furthermore, it is likely that more than one PHB group may be implemented on a node and utilized within a domain. PHB groups should be defined such that the proper resource allocation between groups can be inferred, and integrated mechanisms can be implemented which can simultaneously support two or more groups. A PHB group definition should indicate possible conflicts with previously documented PHB groups which might prevent simultaneous operation.

2.2.12.2 Definition of Differentiated Services Per Domain Behaviors and rules for their specification

Differentiated Services allows an approach to IP Quality of Service that is modular, incrementally deployable, and scalable while introducing minimal per-node complexity [33]. From the end user's point of view, QoS should be supported end-to-end between any pair of hosts. However, this goal is not immediately attainable. It

will require interdomain QoS support, and many untaken steps remain on the road to achieving this. One essential step, the evolution of the business models for interdomain QoS, will necessarily develop outside of the IETF. A goal of the *diffserv WG (Work Group)* is to provide the firm technical foundation that allows these business models to develop.

The first major step will be to support edge-to-edge or intradomain QoS between the ingress and egress of a single network, i.e., a DS Domain in the terminology of [47]. The intention is that this edge-to-edge QoS should be composable, in a purely technical sense, to a quantifiable QoS across a DS Region composed of multiple DS domains.

The Diffserv WG has finished the first phase of standardizing the behaviors required in the forwarding path of all network nodes, the per-hop forwarding behaviors or PHBs. The PHBs defined in [47] [72] and [48] give a rich toolbox for differential packet handling by individual boxes. The general architectural model for diffserv has been documented in [33]. An informal router model describes a model of traffic conditioning and other forwarding behaviors. However, technical issues remain in moving "beyond the box" to intradomain QoS models.

The ultimate goal of creating scalable end-to-end QoS in the Internet requires that we can identify and quantify behavior for a group of packets that is preserved when they are aggregated with other packets as they traverse the Internet. The step of specifying forwarding path attributes on a per-domain basis for a set of packets distinguished only by the mark in the DS field of individual packets is critical in the evolution of Diffserv QoS and should provide the technical input that will aid in the construction of business models.

Diffserv classification and traffic conditioning are applied to packets arriving at the boundary of a DS domain to impose restrictions on the composition of the resultant traffic aggregates, as distinguished by the DSCP marking, inside the domain. The classifiers and traffic conditioners are set to reflect the policy and traffic goals for that domain and may be specified in a TCA (*Traffic Conditioning Agreement*). Once packets have crossed the DS boundary, adherence to diffserv principles makes it possible to group packets solely according to the behavior they receive at each hop (as selected by the DSCP).

This approach has well-known scaling advantages, both in the forwarding path and in the control plane. Less well recognized is that these scaling properties only result if the per-hop behavior definition gives rise to a particular type of invariance under

aggregation. Since the per-hop behavior must be equivalent for every node in the domain, while the set of packets marked for that PHB may be different at every node, PHBs should be defined such that their characteristics do not depend on the traffic volume of the associated BA on a router's ingress link nor on a particular path through the DS domain taken by the packets.

Specifically, different streams of traffic that belong to the same traffic aggregate merge and split as they traverse the network. If the properties of a PDB using a particular PHB hold regardless of how the temporal characteristics of the marked traffic aggregate change as it traverses the domain, then that PDB scales. (Clearly this assumes that numerical parameters such as bandwidth allocated to the particular PDB may be different at different points in the network, and may be adjusted dynamically as traffic volume varies.) If there are limits to where the properties hold, that translates to a limit on the size or topology of a DS domain that can use that PDB. Although useful single-link DS domains might exist, PDBs that are invariant with network size or that have simple relationships with network size and whose properties can be recovered by reapplying rules (that is, forming another diffserv boundary or edge to re-enforce the rules for the traffic aggregate) are needed for building scalable end-to-end quality of service.

There is a clear distinction between the definition of a Per-Domain Behavior in a DS domain and a service that might be specified in a Service Level Agreement. The PDB definition is a technical building block that permits the coupling of classifiers, traffic conditioners, specific PHBs, and particular configurations with a resulting set of specific observable attributes which may be characterized in a variety of ways. These definitions are intended to be useful tools in configuring DS domains, but the PDB (or PDBs) used by a provider is not expected to be visible to customers any more than the specific PHBs employed in the provider's network would be. Network providers are expected to select their own measures to make customer-visible in contracts and these may be stated quite differently from the technical attributes specified in a PDB definition, though the configuration of a PDB might be taken from a *Service Level Specification (SLS)*. Similarly, specific PDBs are intended as tools for ISPs to construct differentiated services offerings; each may choose different sets of tools, or even develop their own, in order to achieve particular externally observable metrics. Nevertheless, the measurable parameters of a PDB are expected to be among the parameters cited directly or indirectly in the Service Level Specification component of a corresponding SLA. [75]

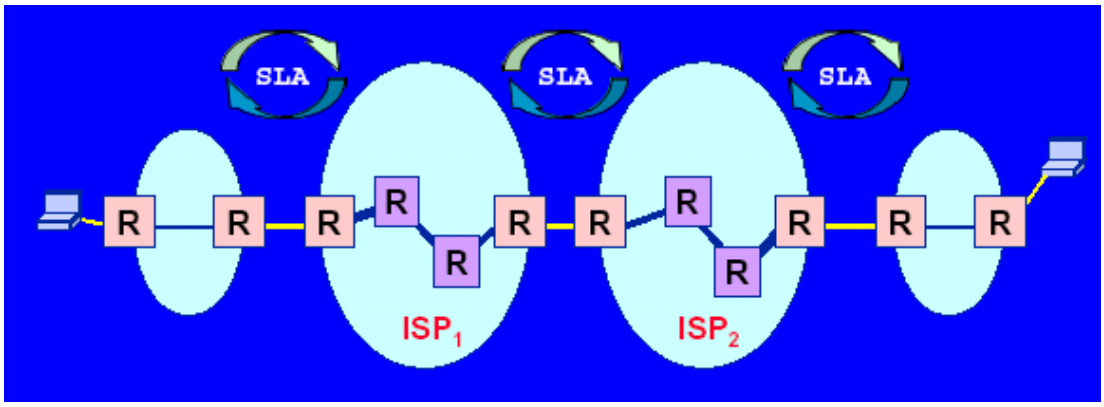


Figure 2.2.12.2.1 Inter-domain Ds

2.2.12.3 The value of defining edge-to-edge behavior

As defined before PDB describes the edge-to-edge behavior across a DS domain's "cloud". Specification of the transit expectations of packets matching a target for a particular diffserv behavior across a DS domain will both assist in the deployment of single-domain QoS and will help enable the composition of end-to-end, cross-domain services. Networks of DS domains can be connected to create end-to-end services by building on the PDB characteristics without regard to the particular PHBs used. This level of abstraction makes it easier to compose cross-domain services as well as making it possible to hide details of a network's internals while exposing information sufficient to enable QoS. Today's Internet is composed of multiple independently administered domains or Autonomous Systems (ASs), represented by the "clouds". To deploy ubiquitous end-to-end quality of service in the Internet, business models must evolve that include issues of charging and reporting that are not in scope for the IETF. In the meantime, there are many possible uses of quality of service within an AS and the IETF can address the technical issues in creating an intradomain QoS within a Differentiated Services framework. In fact, this approach is quite amenable to incremental deployment strategies.

Where DS domains are independently administered, the evolution of the necessary business agreements and future signaling arrangements will take some time, thus, early deployments will be within a single administrative domain. Putting aside the business issues, the same technical issues that arise in interconnecting DS domains with homogeneous administration will arise in interconnecting the autonomous systems (ASs) of the Internet.[75]

2.2.12.4 Per Hop Behavior identification codes

Differentiated Services [47, 33] introduces the notion of Per Hop Behaviors (PHBs) that define how traffic belonging to a particular behavior aggregate is treated at an individual network node. In IP packet headers, PHBs are not indicated as such; instead *Differentiated Services Codepoint (DSCP)* values are used. There are only 64 possible DSCP values, but there is no such limit on the number of PHBs. In a given network domain, there is a locally defined mapping between DSCP values and PHBs. Standardized PHBs recommend a DSCP mapping, but network operators may choose alternative mappings.

In some cases it is necessary or desirable to identify a particular PHB in a protocol message, such as a message negotiating bandwidth management or path selection, especially when such messages pass between management domains. Examples where work is in progress include communication between bandwidth brokers, and MPLS support of diffserv.

In certain cases, what needs to be identified is not an individual PHB, but a set of PHBs. One example is a set of PHBs that must follow the same physical path to prevent re-ordering. An instance of this is the set of three PHBs belonging to a single Assured Forwarding class, such as the PHBs AF11, AF12 and AF13 [72].

2.2.12.5 Usage Scenarios

Diffserv services are expected to be supported over various underlying technologies which we broadly refer to as "link layers" for the purpose of this discussion. For the transport of IP packets, some of these link layers make use of connections or logical connections where the forwarding behavior supported by each link layer device is a property of the connection. In particular, within the link layer domain, each link layer node will schedule traffic depending on which connection the traffic is transported in.

Examples of such "link layers" include ATM and MPLS. For efficient support of diffserv over these link layers, one model is for different Behavior Aggregates (BAs) (or sets of Behavior Aggregates) to be transported over different connections so that they are granted different (and appropriate) forwarding behaviors inside the link layer cloud. When those connections are dynamically established for the transport of diffserv traffic, it is very useful to communicate at connection establishment time what forwarding behavior(s) is (are) to be granted to each connection by the link layer device so that the BAs transported experience consistent forwarding behavior inside the link layer cloud. This can be achieved by including in the connection

establishment signaling messages the encoding of the corresponding PHB, or set of PHBs, as defined in this document.

In another approach, the ATM Forum has a requirement to indicate desired IP QOS treatments in ATM signaling, so that ATM switches can be just as supportive of the desired service as are IP forwarders. To do so, the Forum is defining a new VC call setup information element which will carry PHB identification codes (although will be generalized to do more if needed).

2.2.12.6 Encoding

PHBs and sets of PHBs are encoded in an unsigned 16 bit binary field.

The 16 bit field is arranged as follows:

- **Case 1 :** PHBs defined by standards action, as per [47].

The encoding for a single PHB is the recommended DSCP value for that PHB, left-justified in the 16 bit field, with bits 6 through 15 set to zero. Note that the recommended DSCP value must be used, even if the network in question has chosen a different mapping.

The encoding for a set of PHBs is the numerically smallest of the set of encodings for the various PHBs in the set, with bit 14 set to 1. (Thus for the AF1x PHBs, the encoding is that of the AF11 PHB, with bit 14 set to 1.)

0 1 2 3 4 5	6	7	8	9	10	11	12	13	14	15
DSCP	0	0	0	0	0	0	0	0	X	0

- **Case 2 :** PHBs not defined by standards action, i.e., experimental or local use PHBs as allowed by [47]. In this case an arbitrary 12 bit PHB identification code, assigned by the IANA, is placed left-justified in the 16 bit field. Bit 15 is set to 1, and bit 14 is zero for a single PHB or 1 for a set of PHBs. Bits 12 and 13 are zero.

0 1 2 3 4 5 6 7 8 9 10 11	12	13	14	15
PHB id code	0	0	X	1

Bits 12 and 13 are reserved either for expansion of the PHB identification code, or for other use, at some point in the future. In both cases, when a single PHBID is used to identify a set of PHBs (i.e., bit 14 is set to 1), that set of PHBs must constitute a PHB *Scheduling Class* (i.e., use of PHBs from the set must not cause intra-microflow traffic reordering when different PHBs from the set are applied to traffic in the same microflow). The set of AF1x PHBs [33] is an example of a PHB Scheduling Class.

Sets of PHBs that do not constitute a PHB Scheduling Class can be identified by using more than one PHBID.

2.2.12.7 Signaling the class selector codepoints

[47] defines the eight DS codepoint values of the form 'xxx000' (where x may be '0' or '1') as the Class Selector Codepoints. Codepoint ‘000000’ is the recommended DSCP value for the Default PHB, and hence the Case 1 PHBID constructed from that codepoint is used to signal the Default PHB.

For convenience and consistent operation with networks that employ IP Precedence, the Case 1 format PHBIDs constructed from the other seven Class Selector Codepoints may also be used to signal PHBs. In each case, the PHB signaled by such a PHBID is the PHB to which the embedded class selector codepoint (or IP Precedence value that corresponds to it in non-diffserv domains) is mapped in the recipient's network.

Any specified use of PHBIDs should allow the use of the eight Case 1 PHBIDs constructed from the Class Selector Codepoints.[74]

2.2.12.8 Usage of PHB group

[33] defines a Per-hop behavior (PHB) group to be: "a set of one or more PHBs that can only be meaningfully specified and implemented simultaneously, due to a common constraint applying to all PHBs in the set such as a queue servicing or queue management policy. A PHB group provides a service building block that allows a set of related forwarding behaviors to be specified together (e.g., four dropping priorities). A single PHB is a special case of a PHB group."

One standards track PHB Group is defined in [72], *Assured Forwarding PHB Group*. Assured Forwarding (AF) is a type of forwarding behavior with some assigned level of queuing resources and three drop precedences. An AF PHB Group consists of three PHBs, and uses three Diffserv Codepoints (DSCPs).

[33] defines twelve DSCPs, corresponding to four independent AF classes. The AF classes are referred to as AF1x, AF2x, AF3x, and AF4x (where 'x' is 1, 2, or 3 to represent drop precedence). Each AF class is one instance of an AF PHB Group.

There has been confusion expressed that [33] refers to all four AF classes with their three drop precedences as being part of a single PHB Group. However, since each AF class operates entirely independently of the others, (and thus there is no common

constraint among AF classes as there is among drop precedences within an AF class) this usage is inconsistent with [33]. The inconsistency exists for historical reasons and will be removed in future revisions of the AF specification. It should now be understood that AF is a _type_ of PHB group, and each AF class is an _instance_ of the AF type.

Authors of new PHB specifications should be careful to adhere to the [33] definition of PHB Group. [33] does not prohibit new PHB specifications from assigning enough DSCPs to represent multiple independent instances of their PHB Group. However, such a set of DSCPs must not be referred to as a single PHB Group. [76]

2.2.13 Network resource allocation

The implementation, configuration, operation and administration of the supported PHB groups in the nodes of a DS Domain should effectively partition the resources of those nodes and the inter-node links between behavior aggregates, in accordance with the domain's service provisioning policy. Traffic conditioners can further control the usage of these resources through enforcement of TCAs and possibly through operational feedback from the nodes and traffic conditioners in the domain. Although a range of services can be deployed in the absence of complex traffic conditioning functions (e.g., using only static marking policies), functions such as policing, shaping, and dynamic re-marking enable the deployment of services providing quantitative performance metrics.

The configuration of and interaction between traffic conditioners and interior nodes should be managed by the administrative control of the domain and may require operational control through protocols and a control entity. There is a wide range of possible control models.

Scalability requires that the control of the domain does not require micro-management of the network resources. The most scalable control model would operate nodes in open-loop in the operational timeframe, and would only require administrative-timescale management as SLAs are varied. This simple model may be unsuitable in some circumstances, and some automated but slowly varying operational control (minutes rather than seconds) may be desirable to balance the utilization of the network against the recent load profile [33].

2.2.11.1 Premium Service

This service is not meant to replace the best effort but primarily to meet an emerging demand for a commercial service that can share the network with best effort traffic. It is also called *Expedited Forwarding* (EF) PHB [48]. In contrast with BE traffic which is bursty and requires queue management to deal fairly with congestive episodes, the Premium service, by *design,* creates very regular patterns and small or nonexistent queues. It emulates traditional leased line service that promises to deliver traffic with a low loss probability at a given peak rate. It is best suited for applications requiring low delay and low jitter [16].

2.2.13.2 Assured Service

This model emulates a lightly loaded network even in the presence of congestion. It tries to offer a service that cannot guarantee bandwidth but provides a high probability that high priority tagged packets will be transferred reliably. It corresponds to the *Controlled-Load Service* (of IntServ) and is best suited for applications requiring better reliability than Best-Effort service. Table 2.2.13.1 shows a comparison of the two services [7].

2.2.13.3 Premium and assured services combined

The above services arose from the IETF Munich 1997 meeting in which Dave Clark & Van Jacobson [7][16] each presented work on Diffserv, and each explained how to use one bit of the IP header to deliver a new kind of service to packets in the Internet. Nichols et al [25] exploited the merits that both service can offer and came up with a proposal for a service that permits the use of both these service types a two-bit differentiated services architecture.

Nichols et al. proposed designating two bit patterns from the IP header precedence field. One will be called the Premium or P-bit and the other, the Assured or A-bit. The precedence field is the first three bits of the ToS byte. For this architecture, the border routers perform functions such as marking and classification (in addition to forwarding) on arriving packets whereas intermediate routers need only to implement forwarding functions based on simple priority queuing. The leaf routers are first configured with a traffic profile for a particular flow based on the contents of its header. At the leaf router, all arriving packets have their A and P bits cleared (i.e. reset). The packets are then classified based on the contents of their headers.

If a packet header does not match any configured value, the packet is immediately forwarded. The packets that find a match pass through individual markers that have

been configured from the usage profile for that flow. For an Assured flow, the marker sets the A-bit when the flow is in conformance to its profile, otherwise the flow is unchanged. For a Premium flow, the marker may hold the packets when necessary to enforce their configured rate. Hence premium flow packets emerge from the marker in a shaped flow with their P-bits set.

From the markers, the packets are then passed to the forwarding engines. Packets with their P-bits set enter into a higher priority whereas those that have A-bits set on their headers enter into a lower priority queue with the rest of the ordinary BE traffic. For the border routers, the markers that can be used to implement the two different services can be described by the use of token buckets. For all the routers (both border & intermediate) their output interfaces have two queues and they must implement a test on the P-bit to select a packets output queue. The two queues are serviced by simple priority the premium packets first, whereas the lower priority queue is serviced by the use of RIO mechanism as described in [7].

Table 2.2.13.1. Comparison between Premium Service & Assured Service Models of the DiffServ Architecture

Premium Service	**Assured Service**
Guaranteed peak bandwidth	Depends on how well links are provisioned for bursts.
Placed in high priority queues.	Placed in lower priority queues.
Negligible queueing delay.	Delay characteristics similar to undropped BE traffic.
No buffers needed.	Needs buffer/queue management policies such as RED.
Emulates traditional leased line service.	Emulates a lightly loaded network, even in times of congestion.
Similar to Guaranteed Service of IntServ.	Similar to Controlled-Load Service of IntServ.
Strong appeal for services such as video broadcasts, VoIP, VPN.	Suitable for services that an ISP can provide for individuals (to pay more) for internet services that seem unaffected by congestion periods.

Table 2.2.13.2. Comparison between IntServ & DiffServ Architectures

Integrated Service	Differentiated Service
Per-flow guarantees.	Maps multiple flows into guarantees.
Uses an end-to-end signaling mechanism.	Uses prioritisation based on different classes.
Less scalable because state information increase with number of flows.	More scalable number of flows proportional to number of classes.
Defines two more classes in addition to Best Effort – these are Controlled-Load & Guaranteed Services.	Similar to IntServ in this respect. Additional classes are: Assured Forwarding & Expedited Forwarding (i.e. Premium Service)
In the Internet Network Model, it appears in the Transport Layer .	Appears in the same position in the Internet Network Model as IntServ.

3 ADMISSION CONTROL (AC) IN DIFFSERV

Admission Control will be a key player in the realisation of end-to-end QoS. The QoS mechanisms discussed so far will require that some of the network resources will have to make this decision at some point in time in the network. Admission control implements the decision algorithm that a router or a host uses to determine whether a new flow can be granted the requested QoS without impacting earlier guarantees [27]. The main considerations behind the decision are current traffic load, current QoS, requested traffic profile, requested QoS, pricing, and other policy considerations. For QoS enabled IP networks, Admission Control could be performed in the setting up of RSVP flows or MPLS paths.

Traditional approaches to solve Admission Control problems focused on the requirement of an a priori traffic specification in terms of the *parameters* (such as peak rate, delay) of a deterministic or stochastic model [23]. The admission decision is then based on the specification of the existing flows and the new flow.

This approach, known as *parameter-based*, suffers from several drawbacks, the main one being the inability of the user or application to come up with tight traffic descriptors before establishing the flow [41]. As a result, traffic specification can be expected to be quite loose.

Traditional IP networks only support *best-effort services*, that is, different services with different *QoS* requirements are treated equally by the network, and it is not possible to differentiate among them nor to assure specific QoS targets for a given service.

One of the main elements required in the network to provide QoS is the *Call Admission Control* (*CAC*) mechanism. If the network has no control on the number of flows that are active at the same time, then the overall traffic demand may be higher than the one supported by the network, and the flows may be degraded (e.g. the transmission delay and the percentage of lost packets may be higher than required). With CAC support, once a new flow requests permission to use the network, the admission control algorithm calculates the available bandwidth in each link and decides if there are sufficient resources to admit the new traffic flow while providing the requested QoS.

3.1 Measurement-based admission control (MBAC)

A different approach, in which admission control decisions are made based on network measurements alone, has been the focus of research over the past years. This new approach is called *Measurement-Based Admission Control (MBAC)* and among other things, it alleviates the burden on the users to provide accurate traffic models by shifting the task of traffic specification from the application to the network. The end result is that admission decisions are based on traffic measurements instead of an explicit traffic specification, (i.e. instead of assuming a statistical or worst case model for the traffic).

Tse et al. [41], [23] designed an MBAC scheme in which the basic model used was to consider a bufferless single link with capacity *C*, and that flows arrive over time requesting service. Once admitted, the bandwidth requirement of a flow fluctuates over time while in the system. The assumptions made in this design are that the flow holding time in the system is exponentially distributed with a certain mean, and that departures of the flows are independent of the bandwidth processes.

The MBAC scheme makes decisions of whether to accept or reject a new flow requesting service based on observation of the past traffic flow. Resource overload occurs when the instantaneous aggregate bandwidth demand t_S exceeds the link capacity, *C*; and the QoS was measured by the steady-state overflow probability f_p. The goal of the admission control scheme is to meet a desired QoS objective while maintaining a high average utilisation of the link.

Many different researchers working on this subject area have proposed and worked on a variety of different Admission Control algorithms. Each of these algorithms has two processes [6]: a measurement process that produces an estimate of network load; and a decision algorithm that uses this load estimate to make admission control decisions. Two of such algorithms are briefly described below; each with the two processes mentioned above.

The *Measured Sum* (MS) algorithm [38] admits a new flow if the sum of the token rate of the new flow and the estimated rate of the existing flows is less than a utilisation target *times* the link bandwidth. In this algorithm, the estimated rate of existing flows was derived using a *time window estimator*.

The *Hoeffding Bounds* (HB) [13] admits a new flow if the sum of the peak rate of the new flow and the measured equivalent bandwidth is less than the link utilisation. An *exponential averaging* measurement mechanism is used to produce the load estimate.

The brief descriptions presented above ignore the mathematical and statistical details of the individual algorithms. However, the key point to note is that they differ both in their underlying theory and in the specific measurement and admission control equations they use.

3.2 End-to-End measurement-based admission control (EMBAC)

It has recently been established by *Breslau, Jamin* and *Shenker* that many of the hitherto proposed measurement-based admission controls perform equally well irrespective of the complexity of the predictions they use [6]; and none of them is capable of accurately meeting its loss targets [6]. This lends support to researchers for the simple approach based on measurements between sender and receiver. This new approach to finding solutions to admission control problems is called *End-to-End Measurement-Based Admission Control (EMBAC)* and its basic characterising feature is to rely on end-to-end measures to determine whether there are enough resources to accept a new connection.

3.2.1 The fundamental differences with classical MBAC mechanisms

Firstly, the *decision* on whether to accept or reject a connection is not taken by the internal network nodes/routers but it is independently taken by the edge nodes of each specific connection. Secondly, the *measures* are taken end-to-end by each connection instead of being centrally taken by core routers.

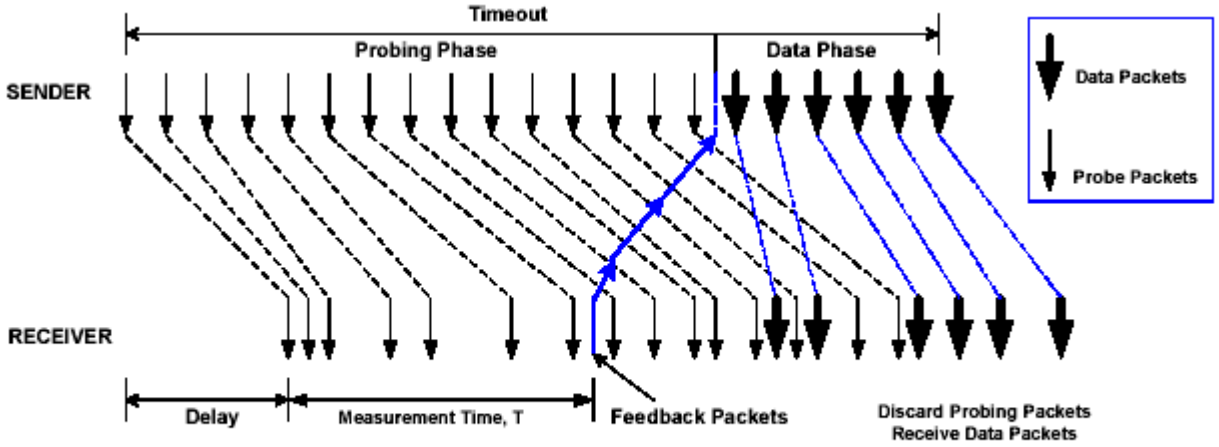


Figure 3.2.1.1 EMBAC Connection setup scheme [3]

As shown in the diagram of figure 3.2.1.1, a connection is composed of two phases: a *probing phase* eventually followed in the case the connection is accepted, by a *data*

phase. First the sender issues a probe; i.e., a stream of packets all of the same length sent at constant inter arrival times**.** Each probe packet includes specification of the transmission rate as well as the session identifier. It may also carry additional information about the session such as specification of the source and channel encodings, but not data. After a certain time interval, the first probing packet arrives at the destination node and the probing packets statistics are measured over a fixed length measurement period *T*.

The destination then sends a measurement report (which consists of the number of probe packets received) to the source. The measurement report is sent with high priority to ensure that it is transmitted with low loss. Based on the admission report, the source decides about the admission and then it either switches from probing to data phase (and starts sending high priority data packets) or aborts the call setup. In the latter case, the source backs off a random time before issuing a new probe.

As an additional possibility, disconnection occurs when a timeout expires. This feature is important in practice when network congestion does not allow any of the probing packets to reach destination. The role of the probing phase is simply to stress the network as much as if the connection were already offered to the network in its data phase. This implies that the probing packet rate must be as large as the additional load generated by the call when accepted. This new mechanism is also known as *Distributed Admission Control* [18].

Also in [12], the controlled-load service [7] with its admission control based on end-to-end measurements is described.

3.3 Measurement-based admission control in egress routers

In this scheme [56], the admission control decisions are only performed by egress routers, without maintaining per-flow state neither in core nor in egress routers. The admission decisions are based only on aggregate measurements collected in the egress router. The key technique is to passively measure the available service in the end-to-end path. Using a *black box* system model, the measurements can incorporate the cross traffic effects without explicitly measuring it or controlling it. Cross traffic is the traffic that is merged in some links with the traffic that is being measured in the egress, but has a different egress router.

For this purpose, the measurement-based theory of envelopes [57] is used to characterize and control both arrivals and services in a general way. Arrival

envelopes are based on the maximum rate of the arrivals. Service envelopes are based on the minimum service available. By measuring the aggregate rate envelope, the short time scale burstiness of the traffic is captured, which is employed in resource reservation and admission control. Then, measuring the variation of the aggregate rate envelope, characterizes the measurement errors at longer time scales, so the variance of the measured envelope can be used to determine the confidence value of the schedulability condition and estimate the expected fraction of packets that a new flow would have in the system if it would be admitted. In the service envelope the cross-traffic effects are measured using the delay of each packet between the ingress and the egress node.

The egress node computes the aggregate arrival envelope and the minimum available service, and then executes the admission control algorithm to accept or deny the new flow. If the minimum available service is sufficient to guarantee a maximum admissible delay for the new flow and to guarantee that the QoS requirements for the already admitted flows are not violated, the flow is admitted.

Disadvantages of Measurement-Based Admission Control in Egress Routers

Although the only router that needs to perform admission control is the egress one, only a large-scale prediction of the congestion level is made. The network conditions may change and the QoS requirements may be degraded.

3.4 End-Point admission control through probing

In this mechanism, [12] the admission of a new flow is performed by the end-hosts or egress/ingress routers through the inference of the network congestion state in the flow's path. Before a new flow is established, the sender sends a packet stream to the flow's path with the same traffic characteristics of the flow that is requesting admission. The packet loss ratio, the delay or delay variation, are measured at the receiver, which verifies the network congestion level. This is called *probing*. If the measured performance is acceptable (according to the required service QoS), the flow is admitted; otherwise it is rejected.

Advantages of End-Point Admission Control through Probing

The QoS functionalities in this mechanism are pushed to the end-points, precluding the need of a signaling protocol or special functions in the core or edge routers.

Disadvantages of End-Point Admission Control through Probing

The overhead introduced with active probing and the set-up time required to initiate a call are some disadvantages of this technique.

3.5 Bandwidth Broker based admission control

Bandwidth Brokers (BB) remove the need for QoS reservation states in the core routers, by centrally storing and managing this information. A BB [59] may be a router or a software package installed in a router/switch in the network.

3.5.1 Main modules of a BB

The main modules of the BB are the call admission control and routing ones. The former maintains the QoS state of the network domain and is responsible for the admission control and resources reservation. The latter decides the path that the admitted flow will traverse towards the receiver. The BB also contains databases with information about network topology, flows and QoS state in each path and node. Usually there is a BB per network domain. Since the sender and receiver can belong to different domains, the BB from their domains and the intermediate ones must communicate the QoS reservation states between each other.

The general description of the call admission control module is as follows: When a new flow with specific traffic parameters, delay and loss requirements requests admission, it sends a QoS request message to the BB. The BB recalculates the available bandwidth in each link, and verifies if there is a path where the new flow can be admitted or not. If the flow is admitted, the BB sends a message to the sender with a positive answer to the flow's request, and updates its database.

3.5.2 General problems in BB

Various BB architectures have been proposed in the recent past, some of them even have been implemented, but all of them are in preliminary stage or do not address important issues such as *bi-directional resource allocation*.

Note that most candidate QoS applications require either bi-directional or such a destination requested flow, e.g. Internet telephony or video-on-demand [53].

3.5.3 An example BB implementation in a DiffServ network

An example BB in a DiffServ network can be seen in Figure 3.5.3.1 Roughly, if user *A* wants to reserve resources for sending data to user *B*, he will send a request to the BB_1. BB_1 checks the *SLA* of user *A* (policy server function) and if authentication and permissions are adequate BB must check resource availability in its domain (call admission control). Since the reservation is initiated for multiple domains in that case, there is a need for communication between bandwidth broker BB_1 and BB_2

*(inter-domain communicatio*n). Note that BB_2 also has to make the call admission control for its own domain to establish end-to-end reservation. Finally, the BB_1 configures the edge routers to classify the user *A* data into the appropriate class.

3.5.3.1 Current BB Implementations

- Qbone
  - Globus: Architecture for Reservation and Allocation (GARA)
  - MCI/WorldCom: Reserved Bandwidth System (RBS)
  - Merit: BB
  - Siemens: QoS-Manager (QoSM)
  - Telia: BB
  - University of Kansas: BB
- Others
  - CKP/NGI BB

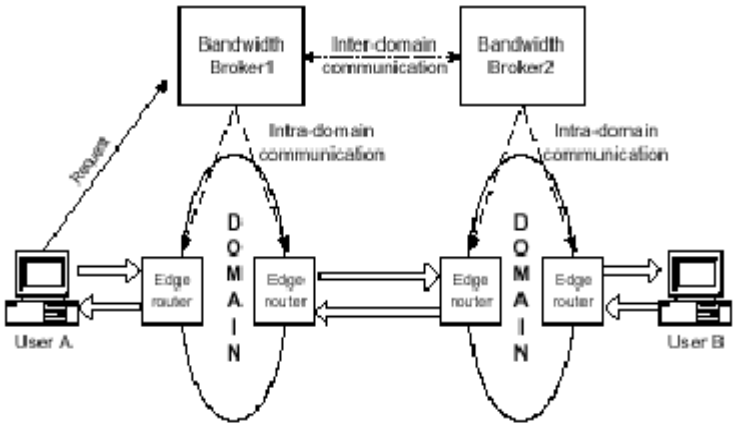


Figure 3.5.3.1 Bandwidth Broker in a DiffServ domain [58]

Advantages of BB based CAC

In this architecture, the core routers will be freed from performing admission control decisions.

Disadvantages of BB based CAC

1. The BB needs to manage the overall network and to store information about all elements, flows and paths in the network. This is very hard for only one element.

2. Therefore, for a large network, a distributed mechanism is preferable.

3.6 Dynamic packet state (DPS)

In the DPS [60] technique, the flow state information (like reserved rate, variables used in the scheduling process) is inserted into packet headers, which overcomes the need for per-flow signaling and state management. The ingress router initializes the state information. Core routers process each incoming packet based on the state (the state can be inserted in four bits of the *Type of Service* (ToS) bytes, which are reserved for experimental use, and into the 13 bits of the *ip_off* field in the IPv4 header, which is used to support packet fragmentation and reassembly carried on it and eventually update its internal state and the state in the packet's header before forwarding it to the next hop.

In terms of admission control, RSVP signaling is used to communicate between the sender and receiver, but RSVP messages are only processed by edge nodes. The ingress node, upon receiving a *PATH* message, simply forwards it through the domain towards the egress node. The egress node, upon receiving the first *RESV* message for a flow, forwards the message to the corresponding ingress node, which in turn will send with a special signaling message along the path towards the egress node. Upon receiving this signaling message, each node along the path performs a local admission control test based on the aggregate reservation rate in that node. When a flow terminates, reservation termination messages are sent in order to release the reserved bandwidth.

Advantages of DPS

With this technique, the core routers are freed from maintaining per flow state.

Disadvantages of DPS

1. A deterministic service is provided since the admission control is based only on the flow's rate inserted in the packet header. This reduces the utilization.

2. Moreover, it is required that all routers in the flow's path implement the same scheduling discipline.

4.1. Differentiated Service types: qualitative services (Implementations and scheduling mechanisms)

4.1.1 Proportional Differentiated Services: delay differentiation and packet scheduling

A different approach in the DS architecture is the *relative differentiated services*. In this approach, the network traffic is grouped into classes of service which are ordered, such that one *Class* is better (or at least no worse) than the other *Class* for, in terms of local (per-hop) metrics for the queueing delays and packet losses [83].
The eight *Class SelectorPHBs*, standardized by the IETF [25], follow the relative service differentiation model. In this context, applications and users do not get an absolute service level assurance, such as an end-to-end delay bound or throughput, since there is no admission control and resource reservations. Instead, the network assures that higher classes will offer better QoS than lower classes, and so it is up to the applications and users to select the class that best meets their requirements, cost, and policy constraints, etc. [26]. Also proportional differentiation model provides the network operator with the *tuning knobs* for adjusting the per-hop quality-of-service (QoS) ratios between classes, independent of the class loads [82].
The paper [80] applies the proportional model in the differentiation of queueing delays, and investigates appropriate packet scheduling mechanisms. Starting from the *proportional delay differentiation (PDD)* model, they derive the average queueing delay in each class, show the dynamics of the class delays under the PDD constraints, and state the conditions in which the PDD model is feasible. The feasibility model of the model can be determined from the average delays that result with the strict priorities scheduler.
They, then focus on scheduling mechanisms that can implement the PDD model in heavy load conditions, when it is feasible to do so. They propose three scheduler. The three schedulers differ in the tradeoff between approximating the PDD model closely, and providing consistent delay differentiation in short timescales.
The *proportional average delay (PAD)* scheduler meets the PDD constraints, when they are feasible, but it exhibits a pathological behavior in short timescales.
The *waiting time priority (WTP)* scheduler, on the other hand, approximates the PDD model closely, even in the short timescales of a few packet departures, but only in heavy load conditions.
PAD and WTP serve as motivation for the third scheduler, called *hybrid proportional delay (HPD)*. HPD approximates the PDD model closely, when the

model is feasible, independent of the class load distribution. Also, HPD provides predictable delay differentiation even in short timescales [82].

4.1.2 A case for relative Differentiated Services and the proportional differentiation model

Internet applications and users have very diverse quality of service expectations, making the same-service-to-all model of the current Internet inadequate and limiting. There is a widespread consensus today that the Internet architecture has to be extended with service differentiation mechanisms so that certain users and applications can get better service than the others at a higher cost.

One approach, referred to as absolute differentiated services, is based on sophisticated admission control and resource reservation mechanisms in order to provide guarantees or statistical assurances for absolute performance measures, such as a minimum service rate or maximum end-to-end delay.

Another approach, which is simpler in terms of implementation, deployment, and network manageability, is to offer relative differentiated services between a small number of service classes.

These classes are ordered based on their packet forwarding quality, in terms of per-hop metrics for the queuing delays and packet losses, giving the assurance that higher classes are better than lower classes. The applications and users, in this context, can dynamically select the class that best meets their quality and pricing constraints, without a priori guarantees for the actual performance level of each class.

The relative differentiation approach can be further refined and quantified using the *proportional differentiation model*. This model aims to provide the network operator with the *tuning knobs* for adjusting the quality spacing between classes, independent of the class loads. When this spacing is feasible in short timescales, it can lead to predictable and controllable class differentiation, which are two important features for any relative differentiation model. The PDM can be approximated in practice with simple forwarding mechanisms (packet scheduling and buffer management) that is described in that paper.

In the absolute differentiation approach, these mechanisms can offer absolute assurances, which are mainly useful for unelastic applications and for deployment scenarios that require specific service measures (e.g., virtual private networks).

In this article, the writers –first- make a case for the relative differentiation approach, as a simply implemented, deployed, and managed solution for service differentiation in the global Internet. They then propose a specific type of relative services, based on the proportional differentiation model. This model allows the network operator to control the quality spacing between classes independent of class loads, and can

provide consistent class differentiation in short timescales. Finally, they describe a packet scheduling (*WTP-Waiting Time Priority*) and buffer management (*Proportional Loss Rate Dropper*) mechanism for approximating the proportional differentiation model in the forwarding engine of a router [83].

4.2 Differentiated Service types: quantitative services (Implementations and Scheduling Mechanisms)

4.2.1 Providing packet loss guarantees in Differentiated Services architectures

Premium services (PS) and *Assured services (AS)* are the first two types of services in DS. PS provides low delay and low jitter service by reserving the peak rate of user flows, thus is more expensive and is used for high-demand applications. But AS provides users with a relatively reliable service, without any quantitative guarantee.
In the paper a new type of service is proposed called *Loss Guaranteed Service (LGS)* for the DS architecture. The LGS can provide a quantitative Qos guarantee in terms of loss rate without per flow based resource reservation. To implement the LGS in DS, a signaling protocol conforming to the Diffserv model, along with a measurement-based admission control is designed. For proposed service model, a simulation model and measurement algorithms in DS architecture developed to study the performance and viability of the model.
Results show that LGS can achieve a high level of utilization while keeping the packet loss within desired level, if the parameters are settled properly. Comparing with PS higher utilization is achieved under the LGS. As a conclusion, LG service can guarantee a loss bound even during congestion periods.[79]

4.2.3 End-to-End QoS guarantees over DiffServ networks

The IntServ uses per-flow signaling and per-flow traffic management, thus introduces significant overhead and scalibility problem. The DiffServ is scalable but can not provide end-to-end service with QoS guarantees by itself. [80] studies various aspects of end-to-end service provisioning over the Internet. Also proposes a new architecture for providing scalable end-to-end QoS, and evaluate the performance of the proposed architecture.
Proposed architectue is based on a signaling protocol over the DiffServ network. The architecture combines per–flow connection setup and per-aggregate traffic handling together. So it guarantees per-flow guarantee during whole network, also scalable. Instead of RSVP a new sender initiated resource reservation protocol is proposed. It is more simple and can be easily implemented. A simulation study is made by using *NS2*. Diffserv implementation module and signaling protocol module are developed.

And various results are given in terms of throughput, end-to-end delay, packet loss rate.

The results of signaling protocol is; the increase of network load both increases end to end delay and packet loss rate of each DS services. *DWRR (Dynamic Weighted Round Robin)* scheduler can provide clear service differentiation between DS traffic classes. Also throughput assurance can be provided by DWRR. Seperating different traffic classes to different queues provide a well-defined fairness of resource allocation between traffic classes [80].

4.3 SIMULATION and software tools for DiffServ network modelling

4.3.1 QUIPS-II : a simulation tool for the design and performance evoluation of Diffserv-based networks

The article in [84] presents a discrete event simulator, called "*Queen's University IP Simulator-II (QUIPS-II)*".

Simulation plays an important role in computer-aided analysis and design of communication networks. QUIPS-II is a such simulation tool written in Java programming language on a UNIX platform. It contains a number of network modules (periodically gathers statistically informations), control modules (for controlling and monitoring the simulation), and a GUI (also writes the performance result to some record files), sharing many features of IETF proposals. The discrete even simulator can be used to investigate and evaluate the behaviour of Diffserv based networks. Also some examples of using QUIPS-II to evaluate the performance of the Premium and Assured Services are given in the paper.

5 REAL TIME APPLICATIONS AND INTERNET

The most important solutions to the real time applications over Internet is *Integrated Services/RSVP* model and *Diffferentiated Services (DiffServ)* model. RSVP is able to provide QoS to each microflow. But it is very difficult to implement on the Internet routers (especially on the backbone routers) because they have to handle hundreds of thousands of flows at the same time. Keeping per flow states in the core routers causes scalibility problems. As stated above, in the DiffServ model packets are marked differently to create several packet classes. The core routers only classify packets based on the packet class instead of the individual micro flow. Core routers do not need to process per flow signaling and resource reservation. So it is relatively easier to implement in the Internet and has better scalibility.

5.1 IP telephony; main advantages and disadvantages

Advantages

1. Internet telephony has been a candidate for the next generation telephone system. Because it uses less bandwidth (hence much cheaper) than traditional circuit switched telephone system [63]. It is more important that the *Public Switched Telephone Networks (PSTN)* toll services can be bypassed using the Internet backbone to reduce the price of long distance calls [64].

2. It is more flexible and it will simplify the physical network.

Disadvantages

Unstable voice quality is the main problem for IP telephony. Because current Internet is designed for the overall transmission throughput and reliability by employing the best effort traffic model. So, TCPIP and best effort services are not suitable for real time traffic models, since they can’t guarantee any bandwidth or delay guarantees.

5.1.1 Structure of the IP telephony Packets

Internet phone is usually delivered by Real-Time Transfer Protocol. RTP lies on top of the UDP protocol.

The telephony audio packets belong to live real time protocol packets (they are more sensitive to latency). Voice is sampled and coded into audio frames. One or more audio frames form an audio packet. Then, RTP headers (time stamp and sequence

numbers) and are added to audio packets to form RTP audio packets. The packet is then sent as a UDP packet to the receiver. At the receiver side, reverse processes are made and the audio packets are played back according to the time stamp in real time. The *International Telecommunication Union (ITU)* has several standards for the IP telephony codecs [63].

The quality of packet voice on the Internet has not been so good because of high packet loss rates. This is especially true on wide-area connections. Unfortunately, the strict delay requirements of real-time multimedia usually eliminate the possibility of retransmissions. It is for this reason that *forward error correction (FEC)* has been proposed to compensate for packet loss in the Internet [65] [66]. In particular, the use of traditional error correcting codes, such as parity, Reed-Solomon, and Hamming codes, has attracted attention. To support these mechanisms, protocol support is required. FEC protocol is independent of the nature of the media being protected, be it audio, video, or otherwise, flexible enough to support a wide variety of FEC mechanisms, designed for adaptivity so that the FEC technique can be modified easily without out of band signaling, and supportive of a number of different mechanisms for transporting the FEC packets [67].

5.1.2 ITU recommendations for voice and video transmission

As stated in [64] and [68], ITU-T Recommendation G.114 specifies that one-way transmission time for connections with adequately controlled echo of voice should be ranging from *0* to *150* ms must be acceptable for most user applications (Some other time ranges are given in Table 5.1.2). However video streaming over IP is able to handle a large jitter margin up to *2s* by using play back buffer at the receiver end.

Table 5.1.2. IP telephony requirements [85].

Quality	Packet Loss (%)	Peak Jitter (ms)
PERFECT	0	0
GOOD	3	75
MEDIUM	10	125
POOR	25	225

5.1.3 Problems (and main solutions) in IP telephony

5.1.3.1 Jitter problem in IP telephony:

Packets may arrive with different end-to-end delay called as *jitter*. Small jitters can be hiden from user by using limited buffer size. But, large buffers are unacceptable, since they introduce long delay.

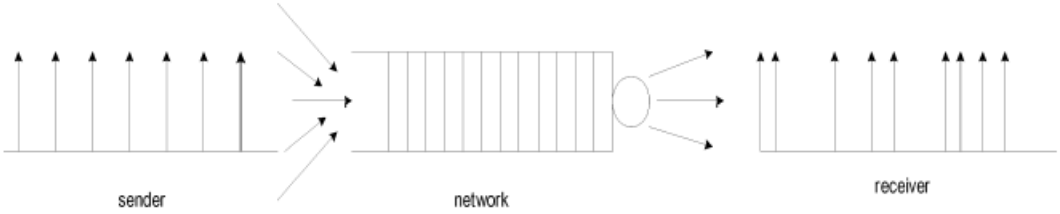


Figure 5.1.3.1 Queueing model for delay jitter [64]

5.1.3.2 Packet losses:

If a packet gets lost or misses the deadline then the receiver will generate a predicted packet based on the neighbouring packets. Retransmission is not acceptable, since it misses the deadline in most cases [63].

5.1.3.3 Bursty nature of TCP traffic:

Another problem for real time IP telephony packets is the bursty nature of TCP traffic. Routers use large size buffers to absorb bursty traffic. So, this poses the problem for real time packets because during congestion, the real time packets will miss their deadlines because of being buffered for extended periods of time. One solution to that problem is mentioned in [44] as; *Techniques like interpolation could be used to tolerate occasional deadline misse*s. Also if a few consecutive audio packets miss their deadlines then, a vital portion of the talk may miss so the quality of the reconstructed audio signal may not be satisfactory [69]. The number of the congested packets depend on the congestion time from several hundreds of miliseconds to several seconds.

The real time services should use a limited buffer size in the routers to limit the total amount of RTP traffic in the core network. Light congestion can still occur in the real time services because there is no strict end to end resource reservation. One way to

avoid following packets be dropped, *selective dropping mechanism* can be used during congestion [63].

5.1.4 Delay types in VoIP call connection

Delay types in VoIP call connection are collected in 3 headlines in [64]:

1. **Accumulation delay:** It is ranging from 25 microsecond to several miliseconds that are caused by the processing of voice samples.

2. **Processing delay:** Is caused by the packetization of voice samples.

3. **Network delay:** Is caused by multiplexing and buffering of voice packets when they are transmitted across networks.

5.1.5 An experiment to see the behaviour of the Internet to the real time packets

In Figure 5.1.8.1 there are three IP telephony streams sent from Iowa State University to the three destinations: China, California and Canada. The experiment was done once every 24 hours a day. Figure 5.1.6.1 shows average lost rate and Figure 5.1.8.2 shows average delay of each call.

5.1.6 Two observations from the experiments

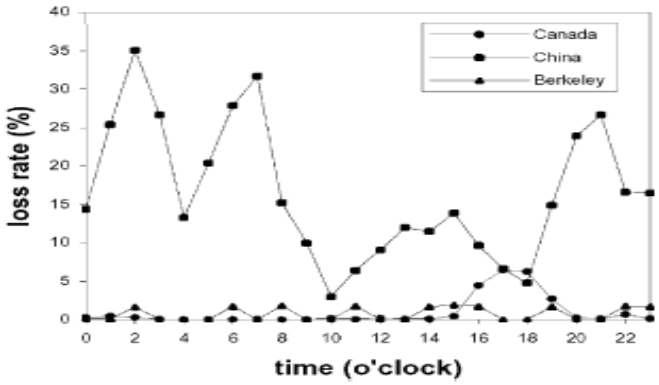


Figure 5.1.6.1 Average loss rate of each connection [70]

1. The overall service quality from ISU to California is pretty good. Also most of the time it is acceptable. But from ISU to China has the worst service quality. Without any technological advancement it can't be used in IP telephony.

2. In the same connection type, the service quality may vary noticeably during different times of a day. During day time the network traffic is heavy so longer

delay and loss rate is higher. However, loss rate is lower in the day time at the stream to China because during day time it is night at China.

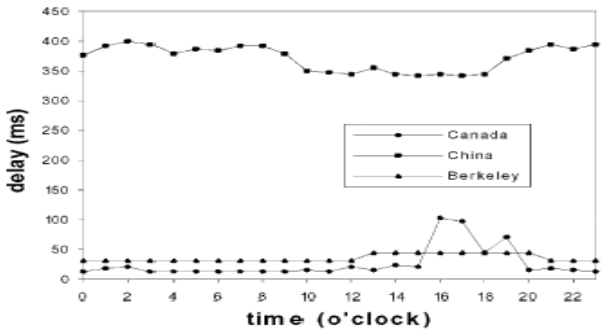


Figure 5.1.6.2 Average packet delay of each connection [70]

5.1.7 Results about the experiments

Average loss rate and the delay of a connection is not sufficient to characterize the voice quality. The voice quality also depends on the distribution of the lost packets, or loss pattern. Packets delayed longer than 200 ms. are accepted as lost packets in the experiment.

5.1.8 (m,k)-firm guarantee

The perceived QoS due to packet losses in a burst is often worse than when the occurences in these losses are adequately spaced. To address this problem a finite horizon QoS criterion called *(m,k)-firm guarantee* was proposed by Ramanathan and Hamdaoui in [70]. A real time stream is said to have an *(m,k)* firm guarantee requirement if *m* out of any *k* consecutive packets in the stream must meet their respective deadlines. In Figure 5.1.10.1, '1' denotes that a packet arrived on time and '0' denotes a packet loss or deadline miss (delay > 200 ms). Consecutive packets of a telephony stream may get lost since it sends multiple packets during a congestion period.

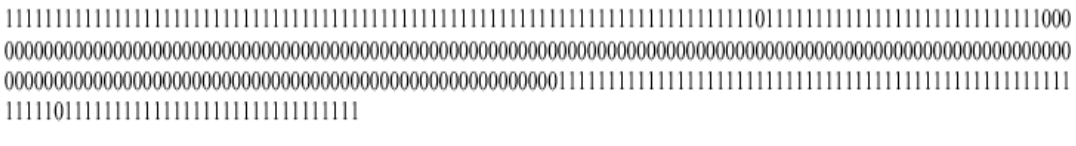


Figure 5.1.8.1 Packet loss during congestion, from ISU to Berkeley, 19 o'clock, CDT [70]

5.2 Congestion difference btw. TCP applications and Internet telephony

Internet telephony is a live real-time process. The service requirement is different from TCP applications. For example, it needs a bounded delay for each packet and a guaranteed bandwidth for the active ‘talk’ period.

TCP is different in nature, uses the sliding window flow control mechanism, which increases the sender rate whenever no congestion is detected. So the network congestion is not avoidable for the TCP traffic.

5.3 Using Differentiated Services to Improve voice quality (CBQ)

If the Internet telephony stream coexists with the TCP streams, during the congestion, some packets may get dropped or delayed. This is not a problem for non real-time TCP streams since they can retransmit the lost packets. But for real time streams, this behaviour is not acceptable because retransmitted packets will miss the deadline in most cases even in short RT time connections as mentioned before. During the congestion period the retransmitted packets are likely to be dropped or delayed again without meeting the deadline. A model can be suggested like in Figure 5.3.1, in the core routers for the real time streams and for the best effort service for the TCP streams.

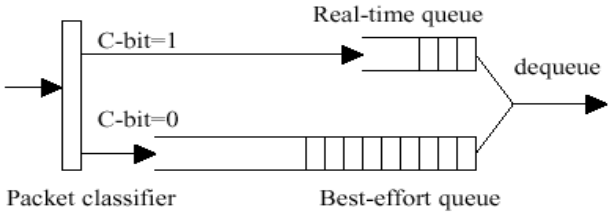


Figure 5.3.1 CBQ model for core routers

5.4 How the model works

Both of the queues are drop tail queues. The buffer size of the real time queue is limited so that a bounded delay could be guaranteed for the real-time services Before a packet enters the Internet from the edge router the one bit of the ToS byte is marked as ‘1’, if the packet is a real time packet and ‘0’ if the packet is a best-effort packet. The packets buffered in the two queues are served using a *Weighted Round*

Robin scheduling schema (instead of priority based servicing in order to avoid starvation of the best effort packets).

A packet dropped by the real time queue is enqueued to the best effort queue. Using this queueing model real time streams are isolated from bursty best–effort streams. Also shaping and policing of the real time traffic are done in the network edge. It is not guaranteed that each real time packet entering the Internet will never get dropped.

However, this can't avoid congestion, as a conclusion in [63]; main aim is to avoid consecutive packet losses.

5.5 Implementation of (m-k) firm guarantee model

In reference [70], Hamdaoui and Ramanathan proposed a *Distance-based Priority assignment* scheme to meet the *(m,k)* firm guarantee QoS requirement of Real Time Streams. The basic idea is that the routers keep the loss history of the last *k* packets for each stream. Based on this history the distance from *(m,k)*-firm guarantee failure is calculated and then a priority is assigned to the current packet based on the distance. The policy is to assign higher priority to streams that are closer to experiencing a failure as defined by the *(m,k)* firm model. The priority can be from 2 levels to 2^n levels. Every 2^n packets from a group and the priority of the packet is the position of that packet in the stream. This priority is assigned at the edge routers. And at the real time queue of the core routers, packets with different priorities are dropped with different probability during the congestion.

5.6 Example queueing model for (m-k) firm guarantee model

Assume that core routers only support 2 priority levels ('0' and '1') in the real-time queue. The queue is a variaton of RED queue. Difference is that, it is configured with two sets of parameters, one for priority '0' and one for priority '1'. The queue works like this:

1. It could drop priority '1' packets earlier than priority '0' packets.
2. Second it drops priority '1' packets with a high probability by setting p_{max1} higher than p_{max0} (marking probability in RED algorithm).

3. Third, if average queue size is greater than MAX (maximum queue size) but smaller than MAX_0, all of the arriving priority '1' packets are dropped, but priority '0' packets may only be dropped with a certain probability (Figure 5.6.1).

By dropping lower priority packets earlier, higher priority packets are transmitted with a lower loss rate. This process is called *selective dropping*. The process can be generalized to more than 2 priority levels. But it could be selected by tradeoff between the queue complexity and the performance improvement.The schema works well for real time streams that have *(m,k)* firm guarantee requirements. However it is difficult to implement this model in the internet.

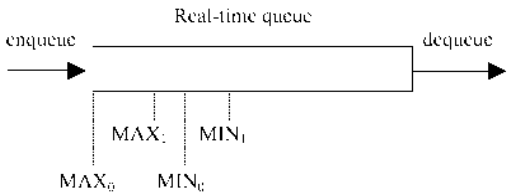


Figure 5.6.1 A selective dropping queueing model

5.7 Advantages of IBO implementation over DB prioritization schema

1. The process is static. A priority is selected before packet enters the Internet.
2. The core routers does not have to keep the loss history of each stream and make the priority assignment decision. Therefore more scalable.
3. It can support an adaptive QoS guarantee based on the network congestion depth.

5.8 Example *(m,k)* firm guarantee models

For example, assuming that the real time queue in the core router could support 8 priority levels, if the loss rate is less than 12.5%, it is (7,8) guarantee. If the loss rate is between 12.5% and 25% it is (3,4)-guarantee. If the loss rate is between 25% and 50% it is (1,2)-guarantee. This is nice for the voice stream because it is hard and necessary to predefine the *m* and *k* in the (m,k)-firm guarantee model. If the network can not satisfy (7,8)-firm guarantee, satisfying (3,4) firm guarantee may also be acceptable.

5.8.1 Main problems of *(m-k)* firm guarantee implementation

1. It needs to keep the loss history of each flow in each router. In the Internet, this will introduce scalebility problems.

2. Routers can only get the local loss history of each stream, they do not know the end-to-end loss history, therefore can't make the global optimum priority assignment.

5.9 Some Conclusions About Real Time Applications in the Internet

1. Since there is no perflow resource reservation in the DiffServ model, the packet loss is unavoidable. Some methods like selective dropping, consecutive packet losses could be avoided during congestion. Also another method can be that each packet can keep some redundant information about the neighboring packet.

2. In the above schema when a packet is lost in a stream than the loss probability of the neighboring packet is low. So redundant information can be used at the receiver side to better predict the lost packet [63].

3. In [71] there is a selective dropping mechanism that can be used also in the transmission of video streams. In an MPEG stream *P* and *B* frames depend on the *I* frame. MPEG video sequences are arranged into *Group of Pictures (GOP)*. Each *GOP* contains three different types of frames:

    - **Intra (I)**: At the beginning of a GOP, an I-frame is transmitted. It is much larger than other frames because complete image is transmitted.

    - **Bi-directional (B)**: After the I-frame, a number of B-frames are transmitted. Has the least number of frames.

    - **Predictive (P)**: P frames are inserted btw. B frames. Require fewer bits than the I-frame.

If an I frame is lost, than the following P and B frames are useless even if they met their deadline. So high priorities can be assigned to the I frames and low priorities to the P and B frames. If the network congestion is not very deep, than the I frames are dropped rarely.

4. Supporting more priority levels in the real time queue could provide finer *(m,k)* firm guarantee. But it is not easy because there is limited buffer size in the real

time queue. Implementing too many RED thresholds in the queue is very difficult [63].

5.10 A simulation study of adaptive voice communications on IP networks

Paper [81] presents simulation results outlining the behaviour of rate-adaptive voice communications over IP networks. In the considered architecture, voice coders adapt their rate to the current state of the network so as to generate only the bandwidth that the network is capable of carrying. An algorithm is proposed for driving the transmission rate of voice sources on the basis of the estimations,according to the network conditions, measured in terms of packet delays and losses.

The effectiveness of the proposed solution is then investigated in such scenarious like:

(i) a dedicated network in which the available bandwidth is exclusively shared between adaptive voice connections;

(ii) a scenario in which adaptive voice sources compete with other TCP-like sources;

(iii) uncontrolled (heterogenous) network environment.

In the proposed architectural solution, variable bit-rate voice coders adapt their rate to the time varying network conditions by means of a control algorithm whose aim is to maximizing the utilization of the available bandwidth while reducing and preventing the occurance of packet losses. Delay and packet loss estimates are employed to drive the algorithm's response to building the congestion. Delay jitter does not play a part in the rate adaptation algorithm, but could be used to devise an algorithm that controls the size of the playout buffer, which is ususally employed to smooth out delay variations.

The results are compared against the performance of non-adaptive (i.e. fixed rate) sources. It is shown that adaptive approach is more effective, being able to carry more voice communications while maintaining an acceptable QoS, even on non-segregated networks [81].

6. A DIFFSERV APPLICATION WITH NS2

6.1 Simulation Environment and Results for Various Queue Types

The topology shown in the Figure 6.1.1 has seven nodes. Three of them are source nodes (node 0, 1 and 2) and one of them is the destination node (node 6). The sources use constant bit rate traffic senders using *UDP* protocol. The traffic sources simulate real time traffic senders. The link between node 4 (core node) and node 5 is the congested link. The capacity of the link between sources and edge node (node 3) is 10 Mb and the link between edge and core node is also 10 Mb. The bandwidth of congested link is 5 Mb (between node 4 and node 5). All link delays are 5 ms. Packet size in the simulation is 1000 Bytes. Simulation period is for 30 seconds long. The rates of the CBR traffic sources are changed from 1 Mb (1.7, 1.8, 2, 2.5, 3 Mb) to 3.3 Mb. The abbreviation *CP* in the result tables are the *code points* showing the packet classes. The *Tx Pkts* is the number of transmitted packets. The *Total Packets* is the total number of the packets that are sent from the source during the simulation period.

All delay measurements are taken between source edge and destination edge. In the source edge, the codepoint and the packet arrival time is put on the each packets header and then this time value is subtracted from the system time in the destination edge. The minimum and maximum delay values of the classes are also given in the tables.

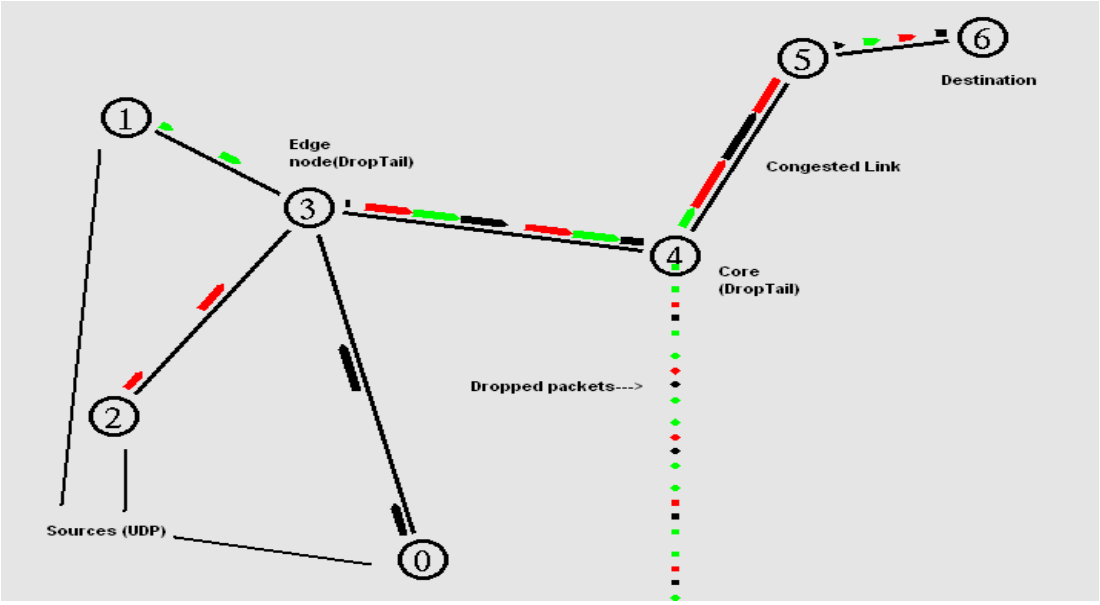


Figure 6.1.1 Non-DiffServ implementation

6.1.1 DiffServ with various queue types

6.1.1.1 DropTail (DT) queue

DT queue sizes are 60 packets long. Packets are dropped in the queue, as the queue overflows. So, there are no QoS guarantee given any of the sources. Each of the sources take the same behaviour from the core and edge routers. All packet drop statistics results for a randomly selected run on the core node are given in the Table 6.1.1.1. Also the average number of dropped packets of 10 random tryings are given at the last column of the table. The starting order of the traffic sources effects the dropped quantity of the packets of the sender. When the total bandwidth consumption of the sources arises the bandwidth of the congested link, then packet dropping starts. This rate limit is 1.7 Mb of each (5.1 Mb total). Also the number of the dropped packets is not stable. It changes randomly. We can see from here that BE service can not give service differentiation to the nodes. It does not give any precedence to any of the sources. Also any of the sources can take enormous amount of the bandwidth. The delay results are given in Table 6.1.1.2. As seen from the charts (also from the Figure 6.1.5) all classes take almost the same delay values. The script code is given in *Appendix 1*.

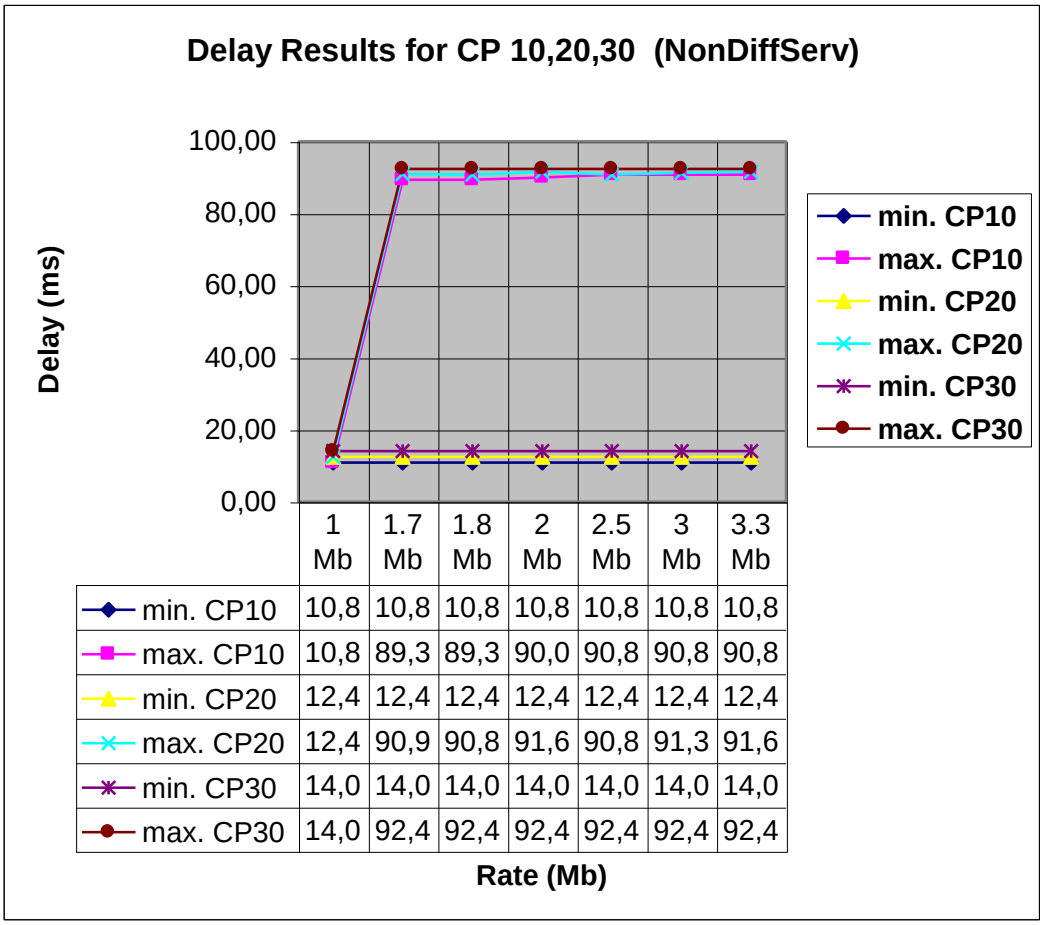
Table 6.1.1.1. Results for NonDiffServ implementation.

Rate	CP	Tot. Pkts	Tx Pkts	Total Dropped	Average dropped #s of 10 runs
1 Mb	**10**	3751	3751	0	**0**
1 Mb	**20**	3751	3751	0	**0**
1 Mb	**30**	3751	3751	0	**0**
1.7 Mb	**10**	6376	6376	0	**0**
1.7 Mb	**20**	6376	6376	0	**0**
1.7 Mb	**30**	6376	6049	327	**326,1**
1.8 Mb	**10**	6750	6750	0	**0**
1.8 Mb	**20**	6750	6750	0	**0**
1.8 Mb	**30**	6750	5299	1451	**1451,4**
2 Mb	**10**	7500	7500	0	**0**
2 Mb	**20**	7500	7499	1	**0,2**
2 Mb	**30**	7500	3800	3700	**3701,2**
2.5 Mb	**10**	9376	9375	1	**958,7**
2.5 Mb	**20**	9376	550	8826	**4279,2**
2.5 Mb	**30**	9376	8877	499	**4088,3**
3 Mb	**10**	11251	7514	3737	**3748,4**
3 Mb	**20**	11251	4006	7245	**6449,8**
3 Mb	**30**	11251	7282	3969	**4752,3**
3.3 Mb	**10**	12375	6392	5983	**5983,9**
3.3 Mb	**20**	12375	6162	6213	**6299,9**
3.3 Mb	**30**	12375	6246	6129	**6042,2**

Table 6.1.1.2. Delay results for nonDiffServ implementation for each classes.

CP	Rate	delay (min)	delay (max)	goodput
10	1 Mb	10.8 ms	10.8 ms	100
20	1 Mb	12.4 ms	12.4 ms	100
30	1 Mb	14.0 ms	14.0 ms	100
10	1.7 Mb	10.8 ms	89.3 ms	100
20	1.7 Mb	12.4 ms	90.9 ms	100
30	1.7 Mb	14.0 ms	92.4 ms	94,87
10	1.8 Mb	10.8 ms	89.3 ms	100
20	1.8 Mb	12.4 ms	90.8 ms	100
30	1.8 Mb	14.0 ms	92.4 ms	78,5
10	2 Mb	10.8 ms	90.0 ms	100
20	2 Mb	12.4 ms	91.6 ms	99,99
30	2 Mb	14.0 ms	92.4 ms	50,67
10	2.5 Mb	10.8 ms	90.8 ms	100
20	2.5 Mb	12.4 ms	90.8 ms	5,866

30	2.5 Mb	14.0 ms	92.4 ms	94,68
10	3 Mb	10.8 ms	90.8 ms	100
20	3 Mb	12.4 ms	91.3 ms	35,61
30	3 Mb	14.0 ms	92.4 ms	64,72
10	3.3 Mb	10.8 ms	90.8 ms	100
20	3.3 Mb	12.4 ms	91.6 ms	49,79
30	3.3 Mb	14.0 ms	92.4 ms	50,47



	1 Mb	1.7 Mb	1.8 Mb	2 Mb	2.5 Mb	3 Mb	3.3 Mb
min. CP10	10,8	10,8	10,8	10,8	10,8	10,8	10,8
max. CP10	10,8	89,3	89,3	90,0	90,8	90,8	90,8
min. CP20	12,4	12,4	12,4	12,4	12,4	12,4	12,4
max. CP20	12,4	90,9	90,8	91,6	90,8	91,3	91,6
min. CP30	14,0	14,0	14,0	14,0	14,0	14,0	14,0
max. CP30	14,0	92,4	92,4	92,4	92,4	92,4	92,4

Figure 6.1.1.1. Minimum and maximum delay values of all classes in the same chart.

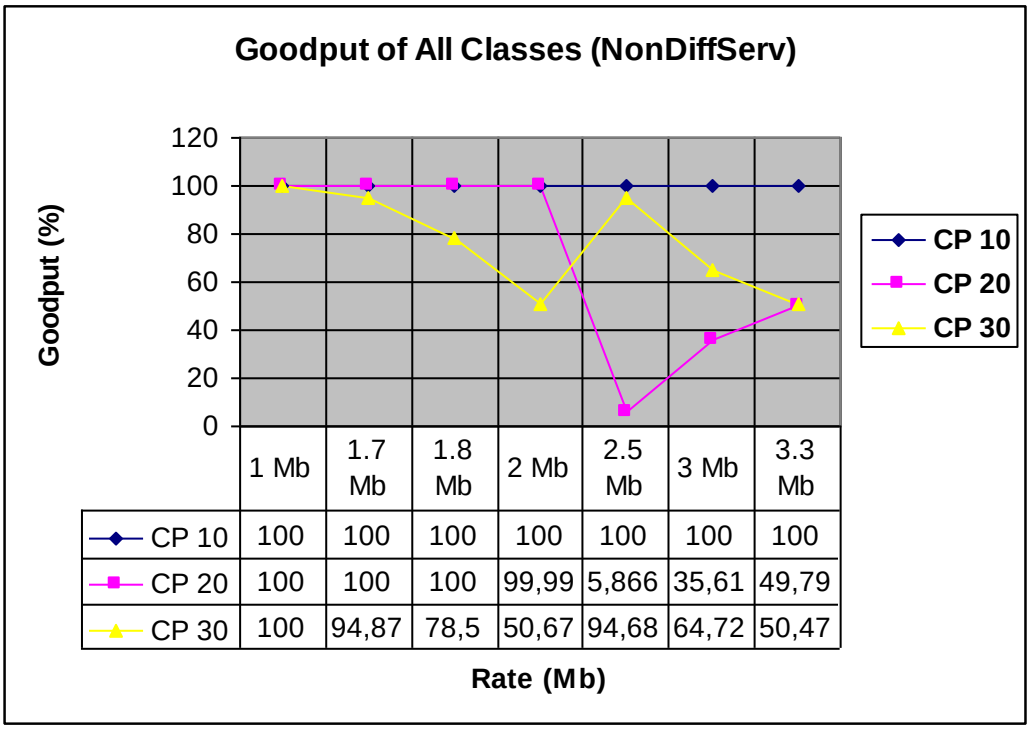


Figure 6.1.1.2 Goodput values of all classes in the same chart.

NonDiffserv implementation with Source Routing

Below, there is a topology with eight nodes. Three nodes are *UDP* sources, one node is the destination node, twos are edge nodes and rest twos are core nodes. Strict source routing is used. In the source routing all path of the packets from source to destination is put on the header of the packets. So each node routes the packets according to that path in the header. Also the routes of three sources are given on the figure. This study is a preparation for the BB implementation. The main disadvantage of source routing is if there is a link down state in the network, packets that use those link are dropped.

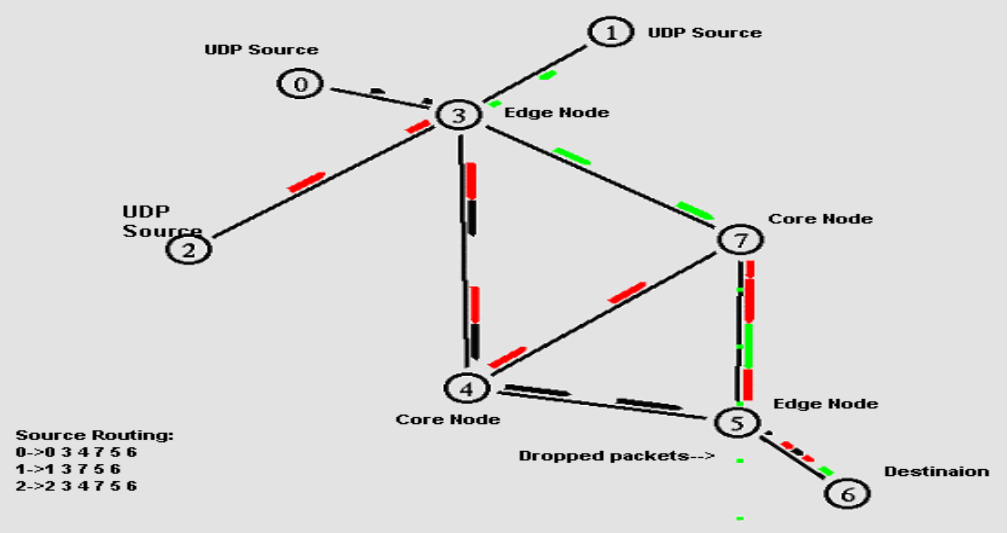


Figure 6.1.1.1.1 Non DiffServ implementation with Source Routing

6.1.1.2 Diffserv with RED queues

In the same topology as in Figure 6.1.1 (except all queues are RED queues) RED queue parameters are given in the Figure 6.1.1.2. There are three physical queues that contains one virtual queue attached to the each physical queue in the topology. Each physical queue is matched with a codepoint. So codepoint 10 is mapped to queue 1, codepoint 20 is mapped to queue 2 and codepoint 30 is mapped to queue 3. The scheduling algorithm is Round Robin. As seen from the Table 6.1.1.2.1, when the total bandwidth consumption of the sources is bigger than 5 Mb the packet dropping occurs. Since the RED parameters of each physical queue are different, the number of dropped packets per class is different that those of others. Classes are sorted high to low. The column *edrops* shows the number of early dropped packets. As shown in the parameters Figure, the dropping probability, minimum and maximum queue sizes become smaller as the codepoint increases and dropping probability increases as the codepoint increases. The delay results are given in the Table 6.1.1.2.2.

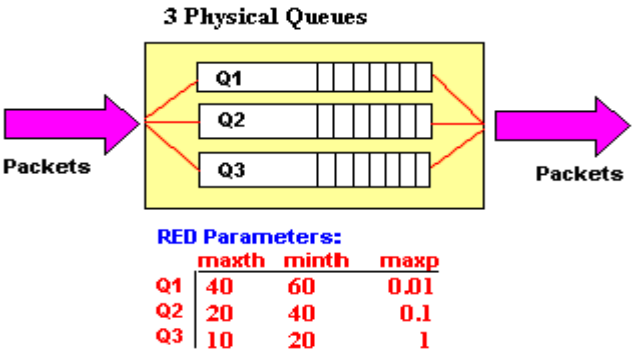


Figure 6.1.1.2.1 RED Queus structure and parameters

Table 6.1.1.2.1. Results for DiffServ Implementation with RED Queues.

Rate	CP	Total Pkts	Tx Pkts	Total Dropped	ldrops	edrops	Average dropped #s of 10 runs
1 Mb	**10**	3751	3751	0	0	0	**0**
1 Mb	**20**	3751	3751	0	0	0	**0**
1 Mb	**30**	3751	3751	0	0	0	**0**
1.7 Mb	**10**	6376	6300	76	61	15	**82,9**
1.7 Mb	**20**	6376	6278	98	0	98	**110**
1.7 Mb	**30**	6376	6259	117	0	117	**127,1**
1.8 Mb	**10**	6750	6304	446	426	20	**491,9**
1.8 Mb	**20**	6750	6288	462	20	442	**508,5**
1.8 Mb	**30**	6750	6258	492	0	492	**543,4**
2 Mb	**10**	7500	6315	1185	1155	30	**1190,2**
2 Mb	**20**	7500	6306	1194	598	596	**1199,2**
2 Mb	**30**	7500	6249	1251	0	1251	**1256,3**
2.5 Mb	**10**	9376	6337	3039	3016	23	**3055,9**
2.5 Mb	**20**	9376	6336	3040	2399	641	**3064,6**
2.5 Mb	**30**	9376	6202	3174	29	3145	**3149,6**
3 Mb	**10**	11251	6320	4931	4912	19	**4935,3**
3 Mb	**20**	11251	6311	4940	4262	678	**4946,4**
3 Mb	**30**	11251	6220	5031	82	4949	**5014,6**
3.3 Mb	**10**	12375	6317	6058	6033	25	**6065,3**
3.3 Mb	**20**	12375	6306	6069	5445	624	**6074,5**
3.3 Mb	**30**	12375	6233	6142	160	5982	**6129,6**

Table 6.1.1.2.2. Delay results for DiffServ implementation with RED Queues.

CP	Rate	delay (min)	delay (max)	goodput
10	1 Mb	10.8 ms	10.8 ms	100
20	1 Mb	12.4 ms	12.4 ms	100
30	1 Mb	14.0 ms	14.0 ms	100
10	1.7 Mb	10.8 ms	250.8 ms	98,81
20	1.7 Mb	12.4 ms	175.0 ms	98,46
30	1.7 Mb	14.0 ms	102.5 ms	98,16
10	1.8 Mb	10.8 ms	250.8 ms	93,39
20	1.8 Mb	12.4 ms	220.0 ms	93,16
30	1.8 Mb	14.0 ms	200.8 ms	92,71
10	2 Mb	10.8 ms	250.8 ms	84,2
20	2 Mb	12.4 ms	251.6 ms	84,08
30	2 Mb	14.0 ms	237.2 ms	83,32
10	2.5 Mb	10.8 ms	250.8 ms	67,59

20	2.5 Mb	12.4 ms	250.8 ms	67,58
30	2.5 Mb	14.0 ms	252.4 ms	66,15
10	3 Mb	10.8 ms	250.8 ms	56,17
20	3 Mb	12.4 ms	251.3 ms	56,09
30	3 Mb	14.0 ms	252.4 ms	55,28
10	3.3 Mb	10.8 ms	250.8 ms	51,05
20	3.3 Mb	12.4 ms	251.6 ms	50,96
30	3.3 Mb	12.5 ms	252.4 ms	50,37

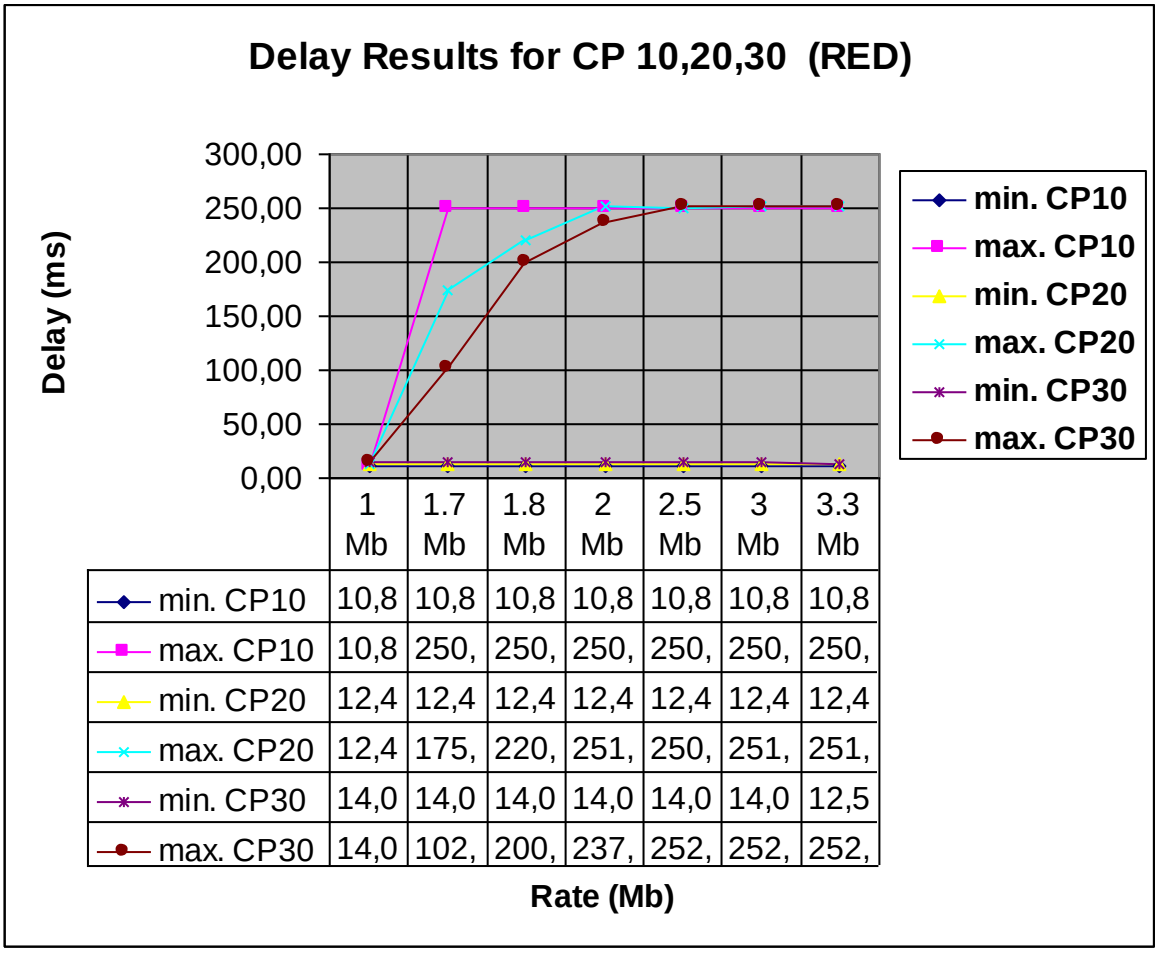


Figure 6.1.1.2.2 Delay values of all classes.

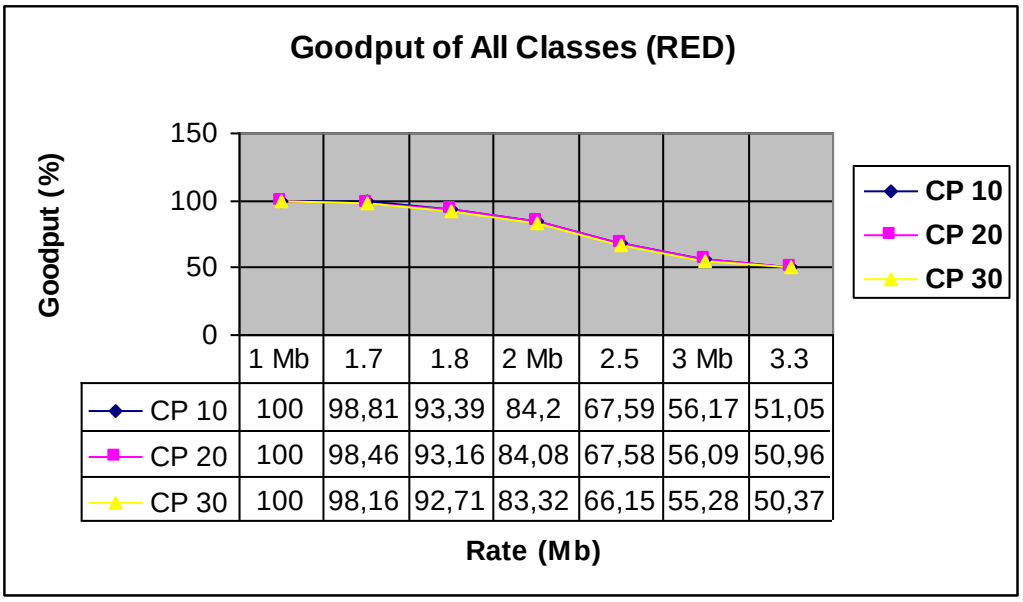


Figure 6.1.1.2.3 Goodput of all classes.

The Figure 6.3.5 shows the delay values of all packets that are reached to the destination. So, because of packet dropping probability of class 30 is more strict, the delay values of that class is better. The class 10 has the worst delay value because of the long queue delays. Because the class 10 packets are not dropped easily. The script code is also given in *Appendix 2*.

6.1.1.3 Diffserv with RED queues (WRR Scheduling)

In the same topology as in Figure 6.1.1 (except all queues are RED queues and the scheduling algorithm for the queues is WRR scheduling algorithm) RED queue parameters are given in the Figure 6.1.1.2. The WRR queue rates are; for CP 10 packets queue it is 7, for the CP 20 packets it is 2 and for the CP 1 packets it is 1. As seen from the Table 6.1.1.3.1, with the WRR scheduling the goodput of the third class increases when using same parameters with the RR schedular. The delay results are given in the Table 6.1.1.3.2. The script code is also given in *Appendix 3*.

Table 6.1.1.3.1. Results for DiffServ Implementation with RED (with WRR Scheduling) Queues.

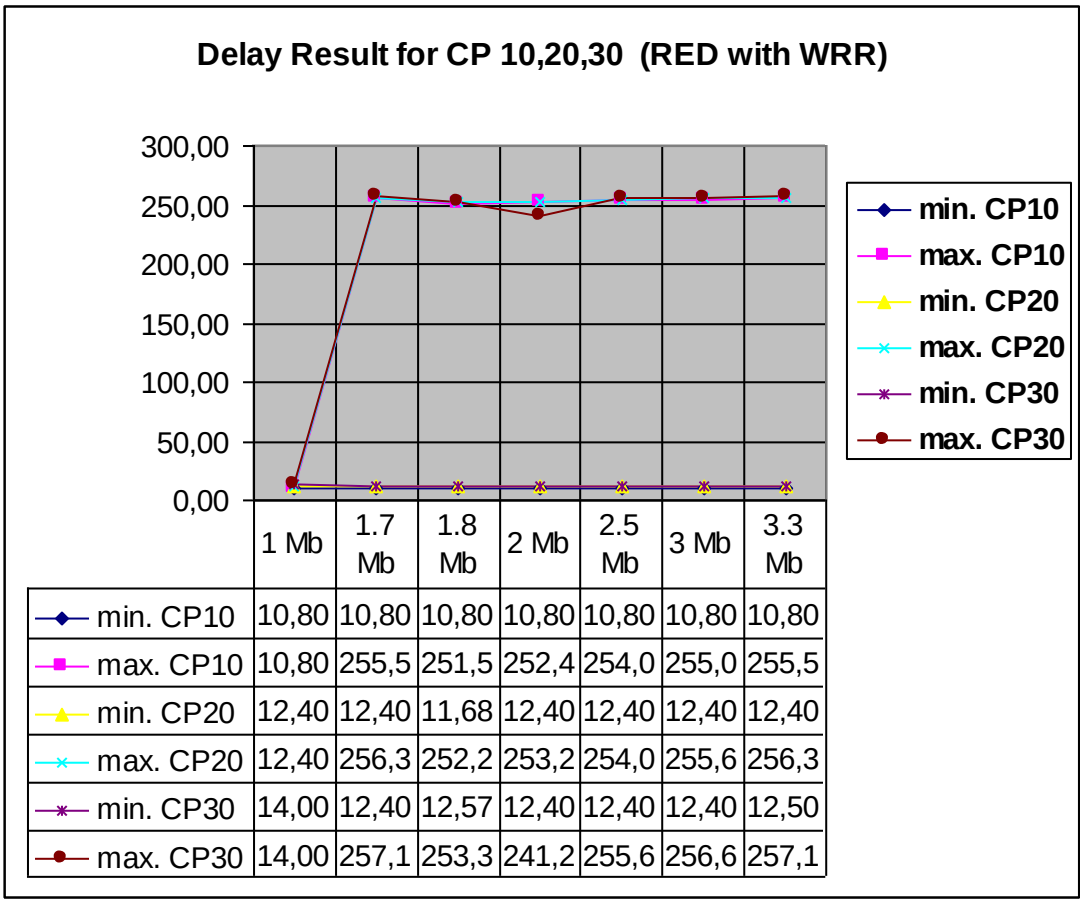
Rate	CP	Total Pkts	Tx Pkts	Total Dropped	Idrops	edrops	Average dropped #s of 10 runs
1 Mb	10	3751	3751	0	0	0	**0**
1 Mb	20	3751	3751	0	0	0	**0**

1 Mb	30	3751	3751	0	0	0	**0**
1.7 Mb	10	6376	6299	289	68	221	**295,3**
1.7 Mb	20	6376	6278	98	0	98	**105,2**
1.7 Mb	30	6376	6259	114	0	114	**122,2**
1.8 Mb	10	6750	6308	442	426	16	**485,9**
1.8 Mb	20	6750	6291	459	13	446	**495,5**
1.8 Mb	30	6750	6236	514	0	514	**553,2**
2 Mb	10	7500	6314	1186	1160	26	**1191,2**
2 Mb	20	7500	6298	1202	623	579	**1207,3**
2 Mb	30	7500	6229	1271	0	1271	**1276,3**
2.5 Mb	10	9376	6341	3035	3015	20	**3055,9**
2.5 Mb	20	9376	6327	3049	2378	671	**3062,6**
2.5 Mb	30	9376	6191	3185	31	3154	**3159,6**
3 Mb	10	11251	6316	4935	4911	24	**4939,3**
3 Mb	20	11251	6304	4947	4318	629	**4951,2**
3 Mb	30	11251	6227	5024	68	4956	**5025,6**
3.3 Mb	10	12375	6311	6064	6036	28	**6070,2**
3.3 Mb	20	12375	6304	6071	5411	660	**6077,4**
3.3 Mb	30	12375	6252	6123	155	5968	**6128,6**

Table 6.1.1.3.2. Delay results for DiffServ implementation with RED (with WRR Scheduling) Queues.

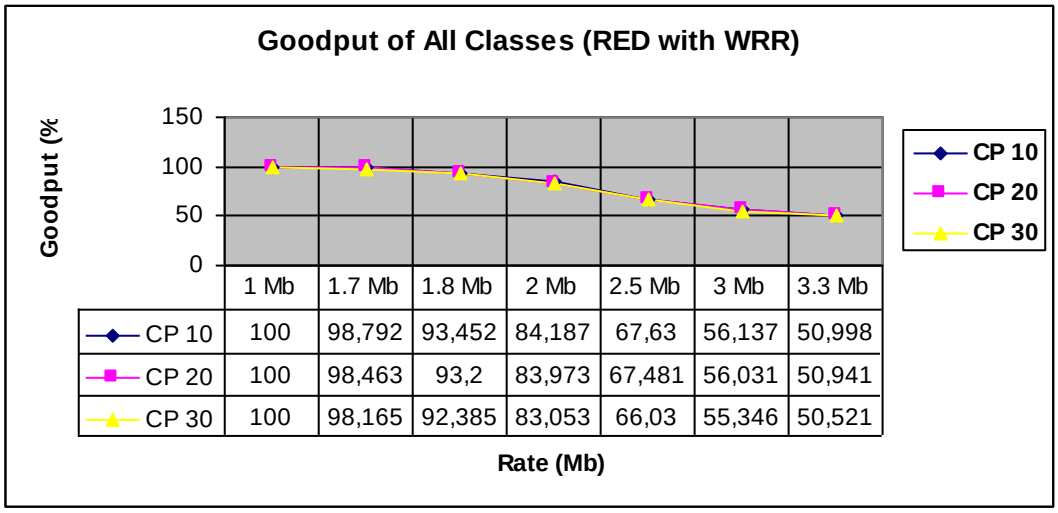
CP	Rate	delay (min)	delay (max)	goodput
10	1 Mb	10,80	10,80	100
20	1 Mb	12,40	12,40	100
30	1 Mb	14,00	14,00	100
10	1.7 Mb	10,80	255,55	98,79235
20	1.7 Mb	12,40	256,32	98,46299
30	1.7 Mb	12,40	257,10	98,16499
10	1.8 Mb	10,80	251,51	93,45185
20	1.8 Mb	11,68	252,22	93,2
30	1.8 Mb	12,57	253,30	92,38519
10	2 Mb	10,80	252,40	84,18667
20	2 Mb	12,40	253,20	83,97333
30	2 Mb	12,40	241,20	83,05333
10	2.5 Mb	10,80	254,00	67,63012
20	2.5 Mb	12,40	254,00	67,4808
30	2.5 Mb	12,40	255,60	66,03029
10	3 Mb	10,80	255,06	56,13723
20	3 Mb	12,40	255,60	56,03058
30	3 Mb	12,40	256,66	55,34619

10	3.3 Mb	10,80	255,55	50,99798
20	3.3 Mb	12,40	256,32	50,94141
30	3.3 Mb	12,50	257,10	50,52121



	1 Mb	1.7 Mb	1.8 Mb	2 Mb	2.5 Mb	3 Mb	3.3 Mb
min. CP10	10,80	10,80	10,80	10,80	10,80	10,80	10,80
max. CP10	10,80	255,5	251,5	252,4	254,0	255,0	255,5
min. CP20	12,40	12,40	11,68	12,40	12,40	12,40	12,40
max. CP20	12,40	256,3	252,2	253,2	254,0	255,6	256,3
min. CP30	14,00	12,40	12,57	12,40	12,40	12,40	12,50
max. CP30	14,00	257,1	253,3	241,2	255,6	256,6	257,1

Figure 6.1.1.3.1 Delay values of all classes.



	1 Mb	1.7 Mb	1.8 Mb	2 Mb	2.5 Mb	3 Mb	3.3 Mb
CP 10	100	98,792	93,452	84,187	67,63	56,137	50,998
CP 20	100	98,463	93,2	83,973	67,481	56,031	50,941
CP 30	100	98,165	92,385	83,053	66,03	55,346	50,521

Figure 6.1.1.3.2 Goodput of all classes.

6.1.1.4 RED with multipriority traffic

We used again the topology in the Figure 6.1.1 with the RED - multipriority traffic with three profiles. RED multipriority parameters are same as with the RED parameters and are given in the Figure 6.1.1.2.1. There are three physical queues in the Figure 6.1.1.2.1, but in the RED – multipriority traffic there are three virtual, one physical queues. In the algorithm, the probability of dropping an out-of-profile packet is based on the weighted average lengths of all virtual queues; while the probability of dropping an in-profile packet is based solely on the weighted average length of its virtual queue. The results of the randomly selected simulation are given in the Table 6.1.1.4.1. If we compare the total dropped column with the same column of RED Table results, the RED-multipriority algorithm gives strict proritization to the packets that are in the CodePoint 10 class at the congestion periods. The script code is also given in *Appendix 4*.

Table 6.1.1.4.1. Results for DiffServ Implementation with RED-multipriority Queues.

Rate	CP	Tot Pkts	Tx Pkts	Total Dropped	Idrops	edrops	Average dropped #s of 10 runs
3 Mb	**All**	11253	11253	0	0	0	
1 Mb	**10**	3751	3751	0	0	0	**0**
1 Mb	**20**	3751	3751	0	0	0	**0**
1 Mb	**30**	3751	3751	0	0	0	**0**
1.7 Mb	**10**	6376	6376	0	0	0	**0**
1.7 Mb	**20**	6376	6376	0	0	0	**0**
1.7 Mb	**30**	6376	6010	366	0	366	**366,7**
1.8 Mb	**10**	6750	6750	0	0	0	**0**
1.8 Mb	**20**	6750	6750	0	0	0	**0**
1.8 Mb	**30**	6750	5253	1497	0	1497	**1497,5**
2 Mb	**10**	7500	7500	0	0	0	**0**
2 Mb	**20**	7500	7500	0	0	0	**0**
2 Mb	**30**	7500	3757	3743	94	3649	**3742**
2.5 Mb	**10**	9376	9376	0	0	0	**5,2**
2.5 Mb	**20**	9376	9268	108	52	56	**62,1**
2.5 Mb	**30**	9376	129	9247	9175	72	**9287,9**
3 Mb	**10**	11251	11229	22	22	0	**6,2**
3 Mb	**20**	11251	7480	3771	2963	808	**3747,9**
3 Mb	**30**	11251	80	11171	11146	25	**11206,5**

3.3 Mb	**10**	12375	12337	38	38	0	**9,6**
3.3 Mb	**20**	12375	6381	5994	5290	704	**5988,4**
3.3 Mb	**30**	12375	74	12301	12278	23	**12337,8**

Table 6.1.1.4.2. Delay results for DiffServ implementation with RED-multipriority Queues.

CP	Rate	delay (min)	delay (max)	goodput
10	1 Mb	10.8 ms	10.8 ms	100
20	1 Mb	12.4 ms	12.4 ms	100
30	1 Mb	14.0 ms	14.0 ms	100
10	1.7 Mb	10.8 ms	41.8 ms	100
20	1.7 Mb	12.4 ms	43.4 ms	100
30	1.7 Mb	14.0 ms	44.9 ms	94,26
10	1.8 Mb	10.8 ms	63.2 ms	100
20	1.8 Mb	12.4 ms	64.8 ms	100
30	1.8 Mb	14.0 ms	66.1 ms	77,822
10	2 Mb	10.8 ms	88.4 ms	100
20	2 Mb	12.4 ms	90.0 ms	100
30	2 Mb	14.0 ms	90.8 ms	50,093
10	2.5 Mb	10.8 ms	90.8 ms	100
20	2.5 Mb	12.4 ms	90.8 ms	98,848
30	2.5 Mb	14.0 ms	92.4 ms	1,3759
10	3 Mb	10.8 ms	90.8 ms	99,804
20	3 Mb	12.4 ms	91.3 ms	66,483
30	3 Mb	14.0 ms	92.4 ms	0,711
10	3.3 Mb	10.8 ms	90.8 ms	99,693
20	3.3 Mb	12.4 ms	91.6 ms	51,564
30	3.3 Mb	14.0 ms	92.4 ms	0,598

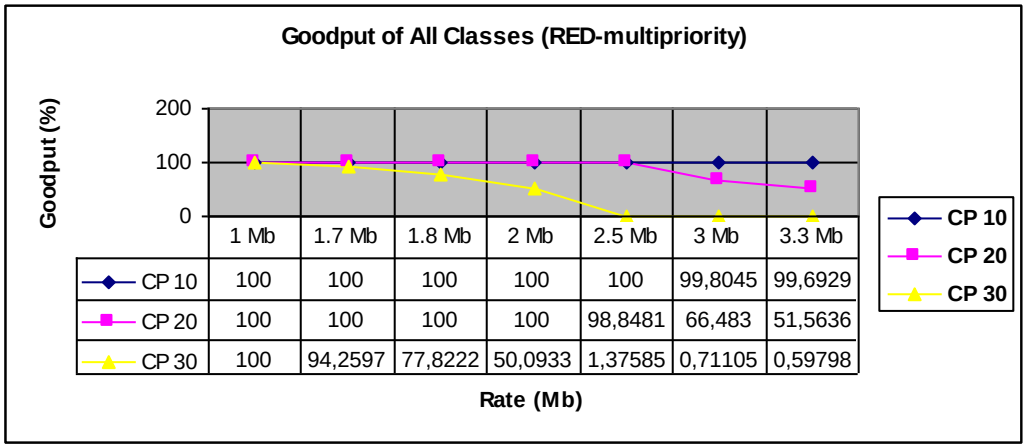
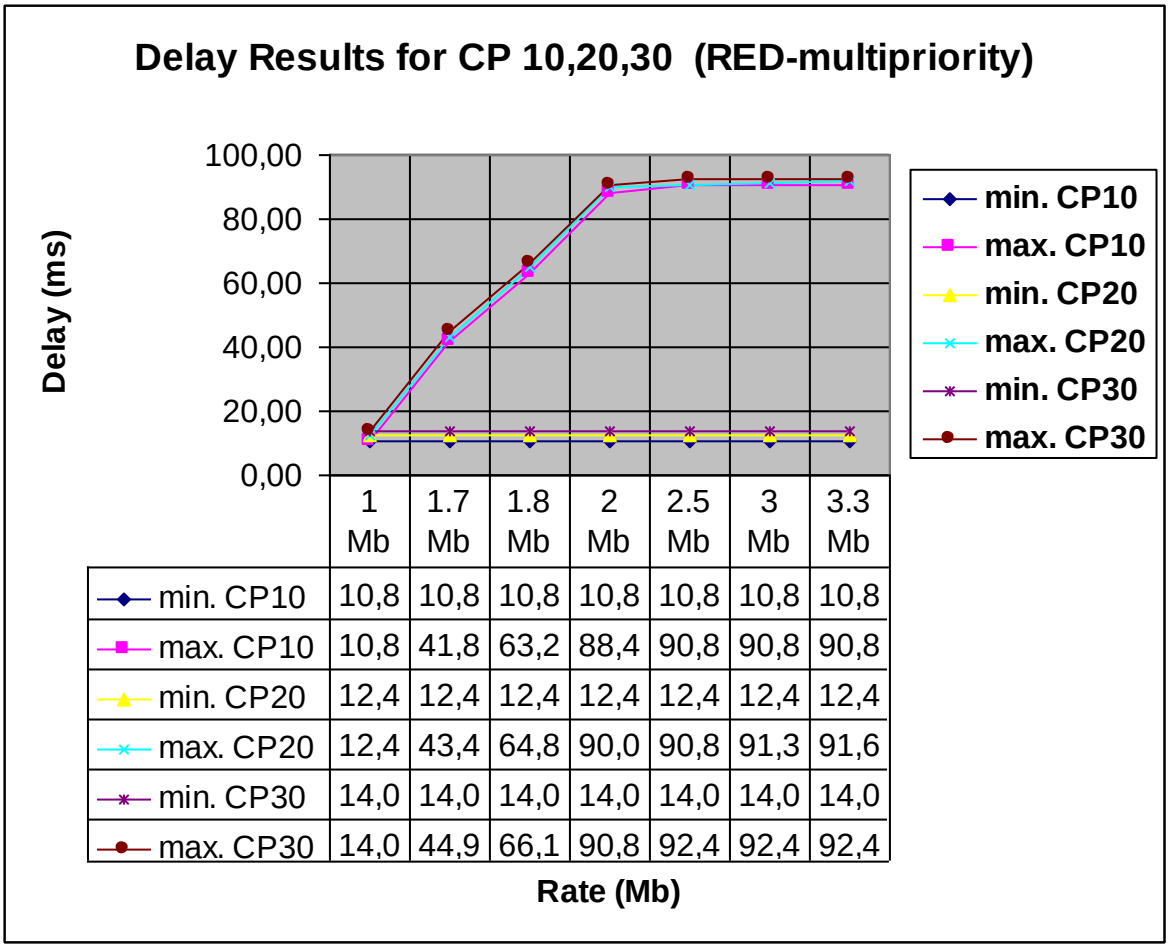


Figure 6.1.1.4.1 Goodput of all classes with RED-multipriority Queues.



	1 Mb	1.7 Mb	1.8 Mb	2 Mb	2.5 Mb	3 Mb	3.3 Mb
min. CP10	10,8	10,8	10,8	10,8	10,8	10,8	10,8
max. CP10	10,8	41,8	63,2	88,4	90,8	90,8	90,8
min. CP20	12,4	12,4	12,4	12,4	12,4	12,4	12,4
max. CP20	12,4	43,4	64,8	90,0	90,8	91,3	91,6
min. CP30	14,0	14,0	14,0	14,0	14,0	14,0	14,0
max. CP30	14,0	44,9	66,1	90,8	92,4	92,4	92,4

Figure 6.1.1.4.2 Delay results of all classes with RED-multipriority Queues in the same chart.

6.1.1.5 Diffserv with PFQ

Again with the same topology as above, all of the queue algorithms are changed to Priority based queue algorithm. In the PFQ (*Priority Fair Queueing*) the router does not drop packets of priority X if there are still packets with priority Y (with X higher than Y). In the below configurations class 10 has the highest priority. The results show that PFQ gives better and strict priorities to the classes than RED algorithm. The script code is given in *Appendix 5*.

Table 6.1.1.5.1. Codepoints and PFQ parameters for DiffServ Implementation with PFQ Queues

	Queue Pri.	Sim Time	Packet Size
CP 10	3	30 secs	1000 B
CP 20	2		
CP 30	1		

Table 6.1.1.5.2. Results for DiffServ Implementation with PFQ Queues.

Rate	CP	Tot Pkts	Tx Pkts	Total Dropped	ldrops	edrops	Average dropped #s of 10 runs
1 Mb	**10**	3751	3751	0	0	0	**0**
1 Mb	**20**	3751	3751	0	0	0	**0**
1 Mb	**30**	3751	3751	0	0	0	**0**
1.7 Mb	**10**	6376	6376	0	0	0	**0**
1.7 Mb	**20**	6376	6376	0	0	0	**0**
1.7 Mb	**30**	6376	6050	326	326	0	**325,1**
1.8 Mb	**10**	6750	6750	0	0	0	**0**
1.8 Mb	**20**	6750	6750	0	0	0	**0**
1.8 Mb	**30**	6750	5300	1450	1450	0	**1450,4**
2 Mb	**10**	7500	7500	0	0	0	**0**
2 Mb	**20**	7500	7500	0	0	0	**0**
2 Mb	**30**	7500	3800	3700	3700	0	**3700,4**
2.5 Mb	**10**	9376	9376	0	0	0	**0**
2.5 Mb	**20**	9376	9376	0	0	0	**0**
2.5 Mb	**30**	9376	51	9325	12325	0	**9325,3**
3 Mb	**10**	11251	11251	0	0	0	**0**
3 Mb	**20**	11251	7546	3705	2942	763	**3708,8**
3 Mb	**30**	11251	50	11201	11201	0	**11200,3**
3.3 Mb	**10**	12375	12375	0	0	0	**0**
3.3 Mb	**20**	12375	6416	5959	5309	650	**5958,2**
3.3 Mb	**30**	12375	50	12325	12325	0	**12325,9**

Table 6.1.1.5.3. Delay results for DiffServ implementation with PFQ Queues.

CP	Rate	delay (min)	delay (max)	goodput
10	1 Mb	10.8 ms	10.8 ms	100
20	1 Mb	12.4 ms	12.4 ms	100
30	1 Mb	14.0 ms	14.0 ms	100
10	1.7 Mb	10.8 ms	10.8 ms	100
20	1.7 Mb	12.4 ms	14.0 ms	100
30	1.7 Mb	14.0 ms	263.6 ms	94,89
10	1.8 Mb	10.8 ms	10.8 ms	100
20	1.8 Mb	12.4 ms	14.0 ms	100
30	1.8 Mb	14.0 ms	342.2 ms	78,52
10	2 Mb	10.8 ms	12.4 ms	100
20	2 Mb	12.4 ms	14.0 ms	100
30	2 Mb	14.0 ms	414.0 ms	50,67

10	2.5 Mb	10.8 ms	12.4 ms	100
20	2.5 Mb	12.4 ms	15.6 ms	100
30	2.5 Mb	36.4 ms	30012.4 ms	0,544
10	3 Mb	10.8 ms	12.4 ms	100
20	3 Mb	12.4 ms	212.4 ms	67,07
30	3 Mb	30033.7 ms	30086.0 ms	0,444
10	3.3 Mb	10.8 ms	12.4 ms	100
20	3.3 Mb	12.4 ms	247.6 ms	51,85
30	3.3 Mb	30036.0 ms	30076.4 ms	0,404

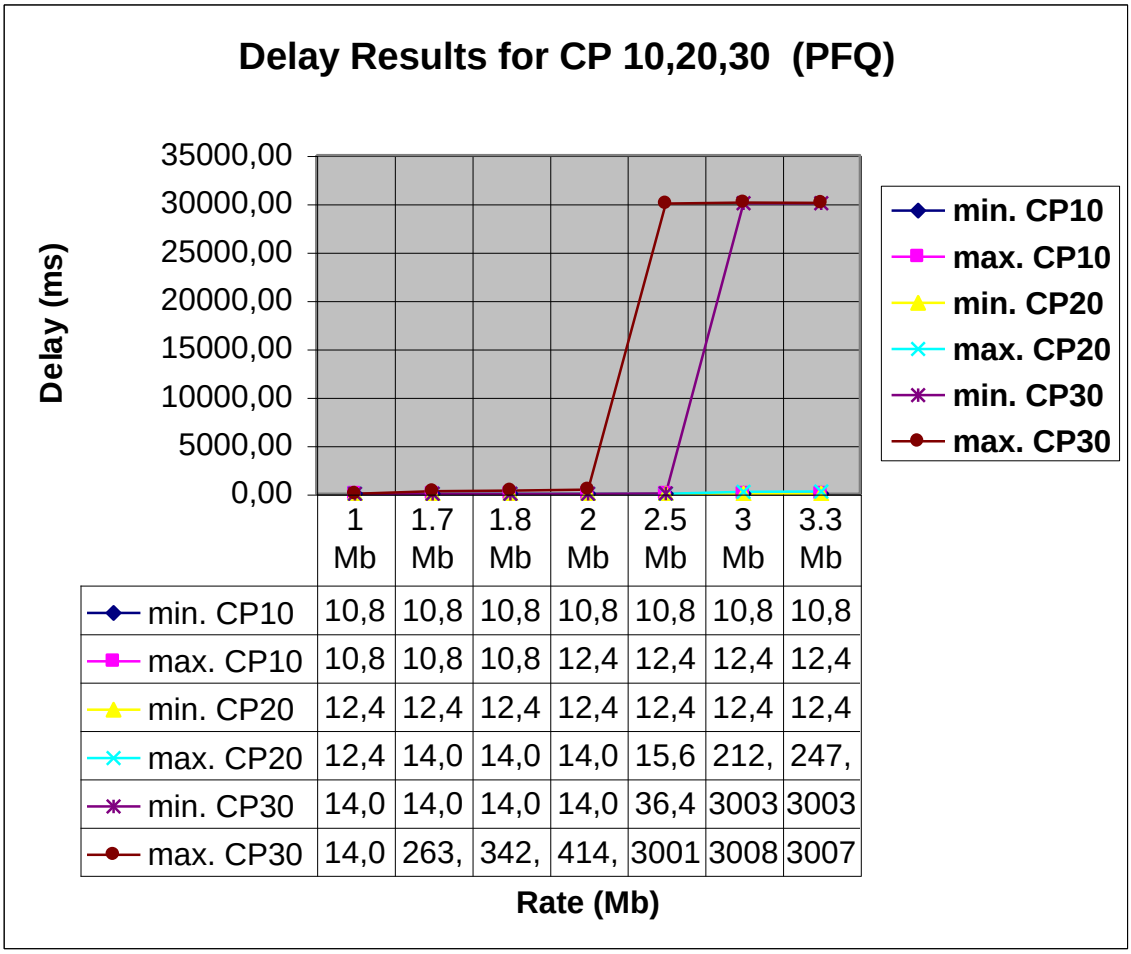


Figure 6.1.1.5.1 Delay results of all classes.

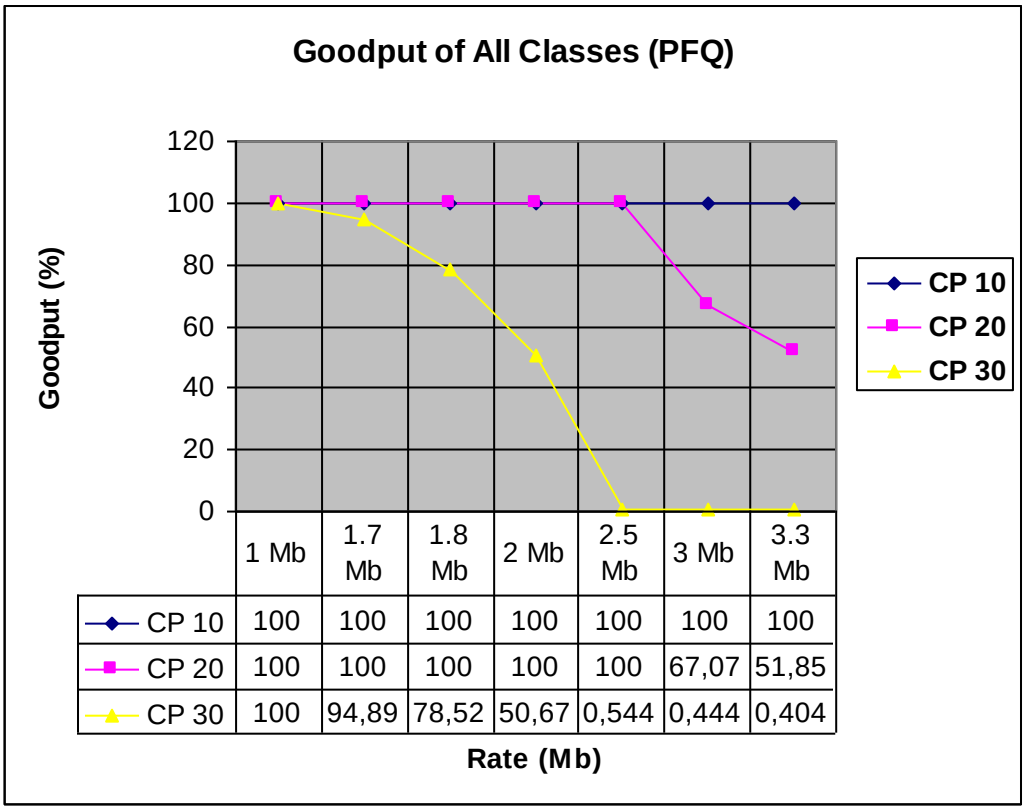


Figure 6.1.1.5.2 Goodput of all classes with PFQ.

6.1.2. Diffserv implementation with Bandwidth Broker

The BB implementation is composed of two main modules: *Admission Control* module and *Routing* Module shown in the Figure 6.1.2.1. The *Dijkstra* algorithm is used to select a path from a source to the destination as shown in Figure 6.1.2.2. The algorithm uses a traffic matrix to find the path of the source that satisfies the traffic requirement of the source. Route metric is the bandwidth required by the traffic source. In the traffic matrix, each row and column show the capacity of the links between the nodes. The link capacity of each link is divided among three classes and the portion per class should be specified as in Table 6.1.2.1 by the BB policy. Like, the Class 1 packets take the 50% of the bandwidth and the Class 2 packets take the 30% of the bandwidth etc.

Table 6.1.2.1. Each cell of the traffic matrix

Nodes	**5**			
1	**Total**	**Class1**	**Class2**	**Class3**
	10	5	3	2

In the topology that is shown in the Figure 6.1.2.2., the BB agent defines the paths of the sources using the *Dijkstra* algorithm with the defined costs according to the traffic descriptors that the sources wanted. The edge node that is shown with node

number 3 is a policer, dropper, marker node. This node marks DS codepoints of the packets with the defined policy rules. And the core nodes only forwards packets according to their codepoints to the defined queues. The script code and the source code of the BB are given Appendix 6 and 7.

In the implemented BB the cost matrix shown in Table 6.1.2.2 is used. In the table all row and columns show the cost in Mb between each nodes. The cells those are in grey color is the total bandwidth between each nodes. Also the area shown in yellow color is used to mention the cost between *ith* and *jth* nodes of the class 10. The other areas are similar to class 20 and class 30. These values determined by the BB after setting the policy (described in 6.1.2.2) of each classes. Like, the class 10 packets take the 50% of the bandwidth and etc.

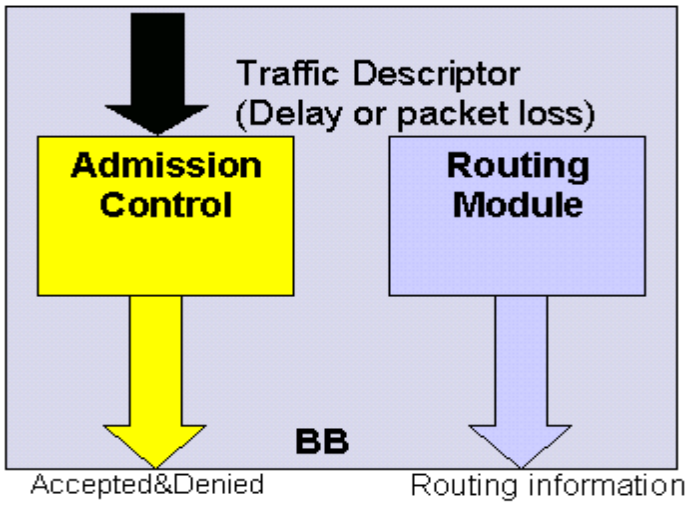


Figure 6.1.2.1 General Modules of a BB

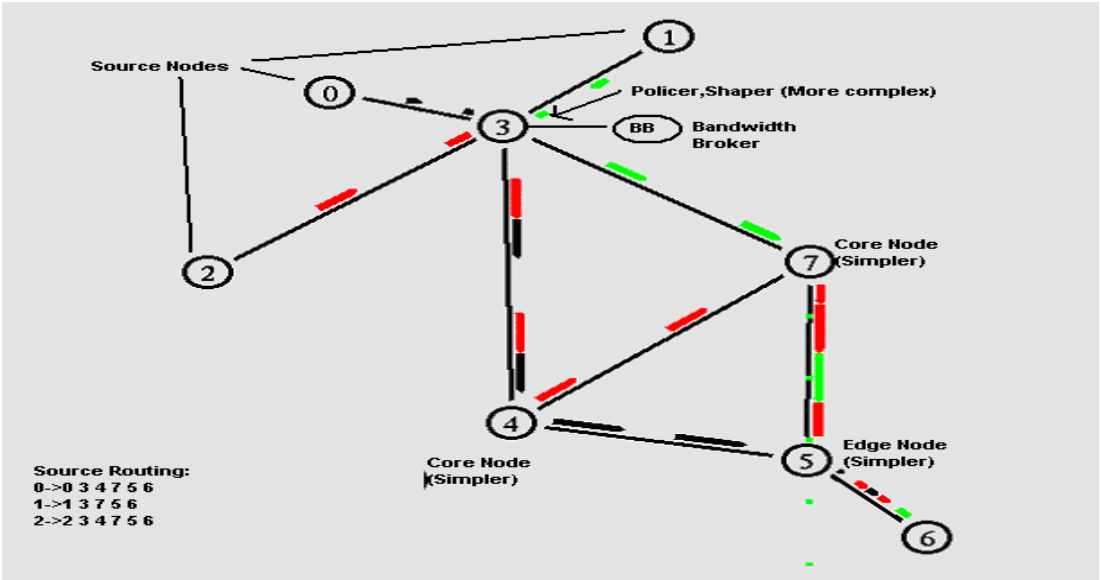


Figure 6.1.2.2 General DiffServ topology with BB

6.1.2.1 General routing strategies

QoS routing perform two tasks: first, collect the state information and keep the traffic matrix up to date. Second, find a feasible path for a new connection according to its service class and traffic descriptor. According to how the state information is maintained and the search of feasible paths is carried out, routing can be divided into three categories as mentioned in [87]:

1. **Source Routing:** As implemented in our applications, source routing achieves simpilicity by transforming a distributed problem into a centralized one (BB), easy to implement, eavulate, debug and upgrade. Disadvantages include communication overhead excessively high for large-scale network, the inaccuracy in the global state may cause the QoS routing failure, computation overhead at the source is excessively high.

2. **Distributed Routing:** The path computation is distributed among the intermediate nodes between the source and destination.

3. **Hierarchical Routing:** The nodes are grouped into multilevel hierarchy. Each physical node maintains an aggregated global state.

6.1.2.2 Policy for each class

The policy used to share the total bandwidth between each link pair is defined by the commands to the BB before the simulation started:

$BB BBPolicy ClassType PercentageOfTotalBandwidth

Such an example can be, *$BB BBPolicy 0 50*. After that command the BB changes the colored yellow of the traffic matrix to the 50% of the cost (colored gray) of all the cells.

6.1.2.3 Link state informations

The routers in the topology send the information of how many packets have passed during the last second. The routers use the formula given below:

*Link usage = ((Cumulative sent packet numbers – Last sent packet numbers) / period) * Packet size * 8 / 1Mb*

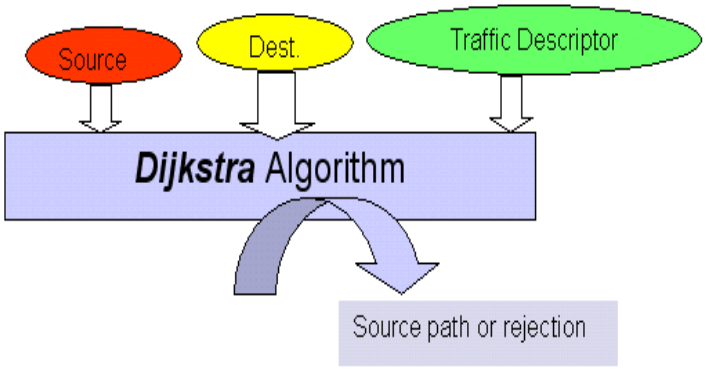


Figure 6.1.2.3 General Dijkstra Algorithm.

Table 6.1.2.2. Traffic matrix of the Routing Module.

	0				**1**				**2**				**3**				**4**				**5**				**6**			
0	0	0	0	0	∞	∞	∞	∞	∞	∞	∞	∞	10	5	3	2	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	∞	∞	∞	∞	0	0	0	0	∞	∞	∞	∞	10	5	3	2	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	∞	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	10	5	3	2	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
3	10	5	3	2	10	5	3	2	10	5	3	2	0	0	0	0	10	5	3	2	∞	∞	∞	∞	10	5	3	2
4	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	10	5	3	2	0	0	0	0	5	2.5	1.5	1	∞	∞	∞	∞
5	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	5	2.5	1.5	1	0	0	0	0	10	5	3	2
6	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	10	5	3	2	0	0	0	0

6.1.2.4 BB with multiple sources

In the topology below, five sources of each class are created to test the BB with multiple sources of each class. To understand the use of the routing module of the BB, one more core node is added to the topology above. The bandwidth between core nodes is also 10 Mb. The capacity of the links between the core nodes to the destination edge node are raised up to the 6 Mb. Also the capacity between destination edge and destination node is raised up to 50 Mb. Bandwidth policy for each class is 60% for class 1 (CP 10), 20% for class 2 (CP 20) and 20% for class 3 (CP 30). In the topology figures, the green colored packets show the class 1 packets, the red colored packets show the class2 packets and the black colored packets show the class 3 packets. Again RED queue algorithm is used at the routers with the same

parameters as above. The simulation duration is about 30 seconds. The rate of each class, CPs, the result of the admission control and the route of the sources are given in the Table 6.1.2.3. The total packet loss results are given in the Table 6.1.2.4. The same topology used without BB and the packet loose results are given in the Table 6.1.2.5. Although the goodput of the first class is 100% in the topology with BB, the goodput becomes 70% in the topology without BB. To show the capability of the AC and routing modules of the BB the configurations in the Table 6.1.2.6. are used and the diagram for that configuration is given in the Figure 6.1.2.5.

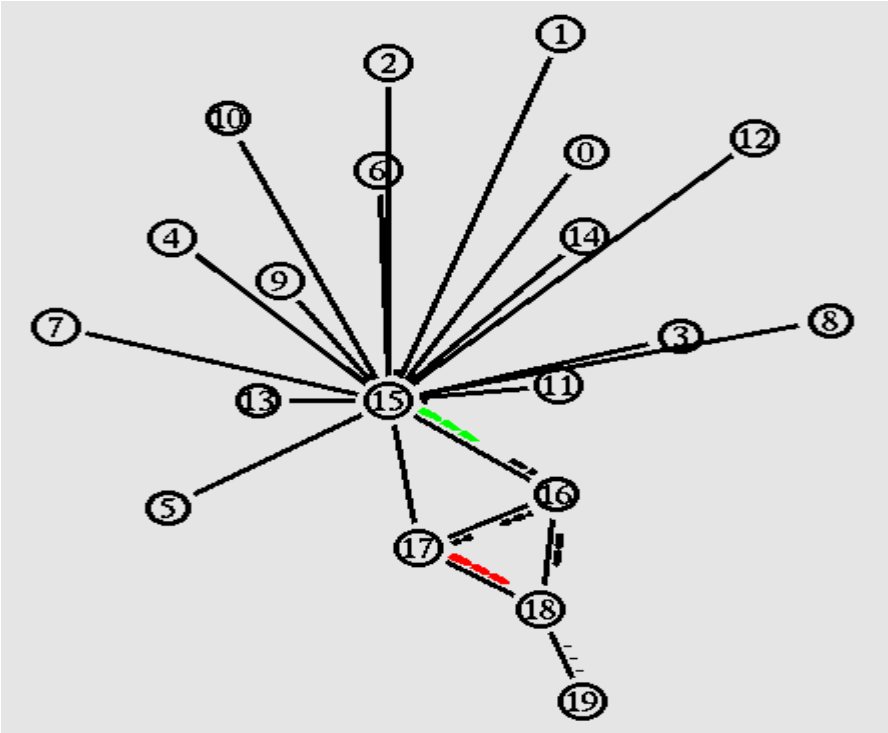


Figure 6.1.2.4 The topology of multiple sources DiffServ network with BB.

Table 6.1.2.3. The results of the AC and routing modules of the BB.

CP (Class)	QoS (Mb)	AC Result	Source	Destination	Route
10	1 Mb	*Accepted*	0	19	0,15,16,17,18,19
10	1 Mb	*Accepted*	1	19	1,15,16,17,18,19
10	1 Mb	*Accepted*	2	19	2,15,16,17,18,19
10	1 Mb	*Accepted*	3	19	3,15,16,18,19
10	1 Mb	*Accepted*	4	19	4,15,16,18,19
20	0.3 Mb	*Accepted*	5	19	5,15,16,17,18,19
20	0.3 Mb	*Accepted*	6	19	6,15,16,17,18,19
20	0.3 Mb	*Accepted*	7	19	7,15,16,17,18,19
20	0.3 Mb	*Accepted*	8	19	8,15,16,17,18,19
20	0.3 Mb	*Accepted*	9	19	9,15,16,18,19
30	0.2 Mb	*Accepted*	10	19	10,15,16,17,18,19
30	0.2 Mb	*Accepted*	11	19	11,15,16,17,18,19
30	0.2 Mb	*Accepted*	12	19	12,15,16,17,18,19

30	0.2 Mb	*Accepted*	13	19	13,15,16,17,18,19
30	0.2 Mb	*Accepted*	14	19	14,15,16,17,18,19

Table 6.1.2.4. Packet loss results for the above topology with BB.

CP	Total	Tx	ldrops	early
10	18744	18744	0	0
20	5625	5625	0	0
30	3752	3752	0	0

Table 6.1.2.5. Packet loss results for the above topology without BB.

CP	Total	Tx	ldrops	early
10	18745	13166	5529	50
20	5625	5625	0	0
30	3750	3750	0	0

Table 6.1.2.6. The results of the AC and routing modules of the BB.

CP (Class)	QoS (Mb)	AC Result	Source	Destination	Route
10	2 Mb	*Accepted*	0	19	0,15,16,17,18,19
10	2 Mb	*Accepted*	1	19	1,15,16,18,19
10	2 Mb	***Not Accepted***	2	19	-
10	2 Mb	***Not Accepted***	3	19	-
10	2 Mb	***Not Accepted***	4	19	-
20	0.6 Mb	*Accepted*	5	19	5,15,16,17,18,19
20	0.6 Mb	*Accepted*	6	19	6,15,16,17,18,19
20	0.6 Mb	*Accepted*	7	19	7,15,16,18,19
20	0.6 Mb	*Accepted*	8	19	8,15,17,16,18,19
20	0.6 Mb	***Not Accepted***	9	19	-
30	0.6 Mb	*Accepted*	10	19	10,15,16,17,18,19
30	0.6 Mb	*Accepted*	11	19	11,15,16,17,18,19
30	0.6 Mb	*Accepted*	12	19	12,15,16,18,19
30	0.6 Mb	*Accepted*	13	19	13,15,17,16,18,19
30	0.6 Mb	***Not Accepted***	14	19	-

The results for the above table are given in the Tables 6.1.2.7 and 6.1.2.8. The tables show that in the network that includes BB, the goodput results are 100%. But in the other hand if there is no BB in the network, the goodput of the classes falls down approximately to 50%s. This shows that BB gives definite service qualities to the sources. The script code of the topology with BB is given in *Appendix 5* and the script code of the topology without BB is given in *Appendix 6*. Also the C++ code of the BB is given in the *Appendix 7*.

Table 6.1.2.7. Packet loss results for the Table 6.1.2.6 with BB.

CP	Total	Tx	ldrops	early	goodput
10	14943	14943	0	0	100
20	8968	8968	0	0	100
30	8967	8967	0	0	100

Table 6.1.2.8. Packet loss results for the Table 6.1.2.6 without BB.

CP	Total	Tx	ldrops	early	goodput
10	14994	7567	7398	29	0,504669
20	11246	7554	2923	769	0,671705
30	11246	7464	29	3753	0,663703

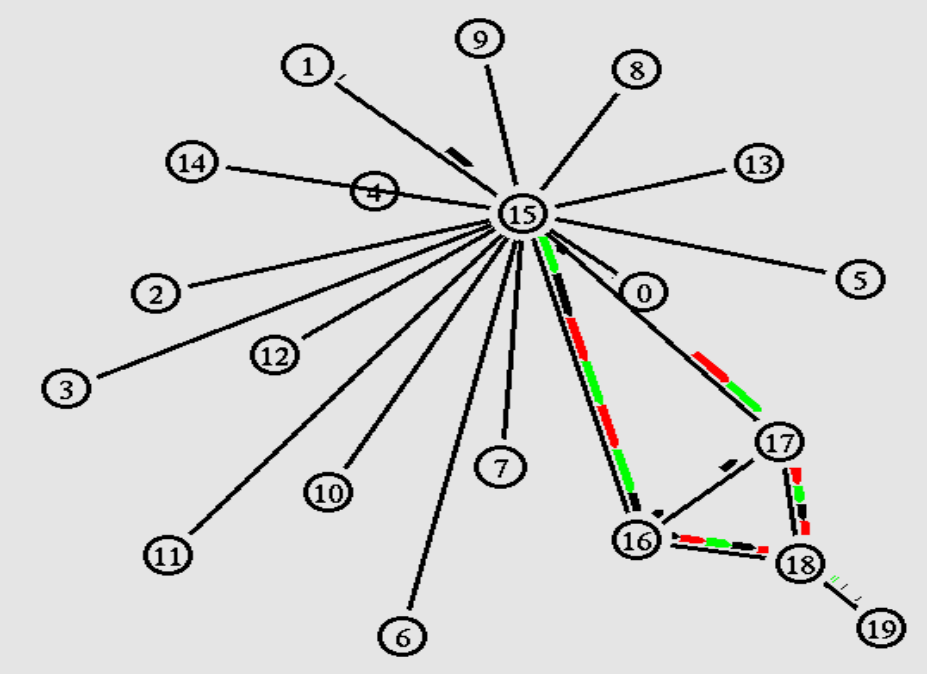


Figure 6.1.2.5 The flows of the classes in the Table 6.1.2.6 with BB.

7. CONCLUSIONS AND FUTURE WORK

Using DiffServ model on edge and core routers creates a QoS enhancement on the special sources. So the packets of those nodes take a better service while traversing the network. But in non-DS environment there is not a service differentiation for the packets of sources. Packets are dropped randomly.

In order to increase QoS in DiffServ environment, the use of BB (*Bandwidth Broker*) is proposed in [80]. In our work, by using the NS simulation tool, we simulated a BB environment by employing various types of queues.

In our studies we used DropTail, RED, PFQ mechanisms in order to manage queues at the routers.

RED queueing algorithm creates service differentiaton on the routers in the IP networks. The selection of RED parameters play important role in the performance of the DS architectures. These parameters must be setted according to the topology, traffic source types, the type and quality of your service.

In the multiple RED case, RR (*Round Robin*) scheduling algorithm is used to extract packets from the RED queues. Another case with the WRR (*Weighted Round Robin*) scheduling is also generated. In the RED with WRR the goodput of the first and third classes are a bit increased.

The results of the PFQ (*Priority Fair Queueing*) algorithm show that the PFQ gives strict priorities to the classes than the RED algorithm. The goodput results of the RED-multipriority and PFQ are similar.

Source routing is a useful way of routing in smaller topologies, but it is not scalable as topology gets larger. In that case, using distributed routing is more scalable.

The BB achieved its keypoint role in supplying QoS to the sources succesfully. In the implementation of BB, Dijkstra algorithm with the QoS requirement which is explained in the thesis is used to find the appropriate path between a source and a destination.

In order to test the performance of the BB, the number of the sources of each class are increased gradually. The results show that implemented BB can achieve the admission control considering the information available at the BB.

Main disadvantage of BB based DiffServ domains is collecting all information on one node (BB). This may cause a problem when that node collapses. Also for the larger topologies, distributed BB mechanisms can be preferred.

In the future, we want to apply active queue management technique BIO, in order to observe the performance of our BB system.

REFERENCES

[1] **P. Aukia,** RATES: *A Server for MPLS Traffic Engineering*, IEEE Network Magazine, Vol **14**, No. 2, pp 207-215.

[2] **Scott M. Ballew**, *Managing IP Networks with Cisco Routers*, O Reilly, First Edition, October 1997.

[3] **G. Bianchi, A. Capone and C. Petrioli**, *Throughput Analysis of End-to-End Measurement-Based Admission Control in IP*, IEEE INFOCOM 2000, vol **3**, page(s) 1461-1470.

[4] **M. May, J. Bolot, A. Jean-Marie and C.Diot**, *Simple Performance Models of Differentiated Services Schemes for the Internet*, INFOCOM.99, 18th Annual Joint Conference of the IEEE Computer & Communications Societies, vol **3**, page(s) 385-1394.

[5] **M. May, T. Bonald and J.Bolot**, *Analytic Evaluation of RED Performance*, INFOCOM 2000, 19th Annual Joint Conference of the IEEE Computer & Communications Societies, vol **3**, page(s) 1415-1424.

[6] **L. Breslau, S. Jamin, S. Shenker**, *Comments on the Performance Measurement-Based Admission Control Algorithms***,** INFOCOM 2000, 19th Annual Joint Conference of the IEEE Computer & Communications Societies, vol **3**, page(s) 1233-1242, March 2000.

[7] **D. Clark & J. Wroclawski**, *An Approach to Service Allocation in the Internet*, Internet Draft draft-clark-diff-svc-alloc-00.txt, July 1997, also talk by D. Clark in the Intserv WG at the Munich IETF, August, 1997.

[8] **C. Metz**, *RSVP: General-Purpose Signaling for IP., EEE Internet Computing*, May-June, 1999, vol **3**, no. 3, page(s) 95-99.

[9] **Douglas E. Comer**, *Internetworking with TCP/IP, Principles, Protocols & Architecture*, vol **1**, 2nd Edition, 1991.

[10] **S. Crosby, I. Leslie, B. McGurk, J. T. Lewis, R. Russel and F. Toomey**, *Statistical Properties of a near-optimal measurement-based CAC algorithm*, In Proceedings IEEE ATM .97, June 1997.

[11] **S.N. Diggavi and M. Grossglauser**, *Information Transmission over a Finite Buffer Channel*, IEEE Proceedings, International Symposium on Information Theory 2000, page(s) 52-52.

[12] **V. Elek, G. Karlsson and R. Ronngren**, *Admission Control Based on End-to-End Measurements*, INFOCOM 2000, 19th Annual Joint Conference of the IEEE Computer & Communications Societies, Vol. **2**, pp 622-630, March 2000.

[13] **S. Floyd**, *Comments on Measurement-Based Admission Control Algorithms for Controlled Load Services*, Technical Report, Lawrence Berkley Laboratory, July 1996.

[14] **R. Gibbens, F. Kelly**, *Measurement-Based Connection Admission Control*, 15th, International Teletraffic Congress, June 1997.

[15] **Bassam Halabi**, *Internet Routing Architectures*, Cisco Press, 1997.

[16] **V. Jacobson**, *Differentiated Services Architecture*, talk in the IntServ WG at the Munich IETF, August, 1997.

[17] **G. Karlsson, F. Orava**, *The DIY approach to QoS, in Proc. IWQOS*, pp 6-8, London, UK, May 1999.

[18] **F.P. Kelly, P.B. Key and S. Zachary**, *Distributed Admission Control*, IEEE Journal on Selected Areas in Communications, 18:(12), December 2000, pp. 2617-2628.

[19] **E. W. knightly, J. Qiu**, *Measurement-Based Admission Control with Aggregate Traffic Envelopes*, In IEEE ITWDC 98, (Ischa, Italy, Setp.1998).

[20] **L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas**, *LDP Specification*, IETF Internet Draft, <draft-ieft-mpls-ldp-08.txt>, June 2000.

[21] **C. Metz**, *IP Over 2000: Where Have We Been And Where Are We Going?*, IEEE Internet Computing, Jan-Feb, 2000, vol **4**, no. 1, page(s) 83-87.

[22] **C. Metz**, *IP Qos:Travelling in First Class on the Internet., IEEE Internet Computing*, March-April, 1999, vol **3**, no. 2, page(s) 84-88.

[23] **M. Grossglauser, D. N. C. Tse**, *Measurement-Based Call Admission Control: Analysis & Simulation*, IEEE 1997

[24] **M. Grossglauser and D.N.C. Tse**, *A Time-scale Decomposition Approach to Measurement-Based Admission Control*, IEEE INFOCOM 99, vol **3**, page(s) 1539-1547.

[25] **K. Nichols, V. Jacobson, L. Zhang**, *A Two-bit Differentiated Services Architecture for the Internet*, Intenet Draft, <draft-nichols-diff-svc-arch-02>, April, 1999.

[26] **S. Nilsson and G. Karlsson**, *Fast Address Lookup for Internet Routers*, ACM SIGCOMM 97, Cannes, France, Sept.1997 http://www.acm.org./sigcomm/sigcomm 97

[27] **R. Branden, D. Clark and S. Shenker**, *Integrated Services in the Internet Architecture: An Overview*, RFC 1633, June 1994.

[28] **R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin**, *Resource Reservation Protocol (RSVP)*, Version 1, Functional Specification, RFC 2205, September 1997, http://www.ietf.org/rfc/rfc2205.txt

[29] **J. Wroclawski**, *Specification of the Controlled-Load Network Element Service*, RFC 2211, September 1997.

[30] **S. Shenker, C. Patdridge and R. Guerin**, *Specification of Guaranteed Quality of Service*, RFC 2212, September 1997.

[31] **S. Shenker and J. Wroclawski**, *General Characterisation Parameters for Integrated Service Network Elements*, RFC 2215, September 1997.

[32] **B. Braden**, *Recommendation on Queue Management and Congestion Avoidance in the Internet*, RFC 2309, April, 1998.

[33] **S. Blake**, *An Architecture for Differentiated Services*, IETF RFC 2475, Dec 1998.

[34] **K. Ramakrishnan, S. Floyd**, *Explicit Congestion Notification (ECN)*, RFC 2481, January 1999.

[35] **S. Sahu, D. Towsley and J. Kurose**, *A Quantitative Study of Differentiated Services for the Internet*, Global Telecommunications Conference 1999, GLOBECOM 99, vol **3**, page(s) 1808-1817.

[36] **J. Saltzer, D.Reed and D. Clark**, *End-to-End Arguments in System Design, ACM Transactions in Computer Systems*, Nov 1984, http://www.reed.cCom/ papers/endtoend.html

[37] **Mischa Schwartz**, *Telecommunication Networks, Protocols, Modelling & Analysis*, Addison-Wesley, 1987.

[38] **S. Shenker, S. Jamin, P. Danzig**, *Comparison of Measurement-Based Admission Control Algorithms for Controlled Load Service*, Proceedings of the Conference on Computer Communications (IEEE Infocom), April, 1997.

[39] **Andrew S. Tanenbaum**, *Computer Networks*, 3rd Edition, Prentice Hall, 1996.

[40] **B. Teitelbaum, S. Hares, L. Dunn, V. Narayan, R. Neilson, F. Reichmeyer**, *Internet2 Qbone-Building a Test bed for Differentiated Services*, IEEE Network, vol.**13**, No.5, Sept/Oct. 1999.

[41] **M. Grossglauser and D.N.C. Tse**, *A Framework for Robust Measurement-Based Admission Control*, IEEE/ACM Transactions on Networking, June 1999, vol **7**, no. 3, page(s) 293-309.

[42] **M. Waldvoge, G. Varghese, J. Turner & B.Plattner**, *Scalable High Speed IP Routing Lookups*, ACM SIGCOMM 97, Cannes, France, Sept. 1997. http://www.acm.org/sigcomm/sigcomm97

[43] *QoS Protocols and Architectures*, White Paper, http://www.qosforum.com.

[44] **X. Xiao and L.M. Ni**, *Internet QoS: The Big Picture*, IEEE Network 1999, vol **12** iss 2, pp8-18.

[45] Lecture notes, *Computer Networking-IntServ, RSVP and DiffServ*

[46] Request for Comments: 791, STANDARD, J. Postel, Sep-01-1981, *Protocol Internet*

[47] Request for Comments: 2474, Network Working Group, **K.Nichols** (Cisco Systems), Obsoletes: 1455, 1349, **S. Blake** (Torrent Networking Technologies), **F. Baker** (D. Black), December 1998, Category:

Standards Track Cisco Systems EMC Corporation *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*

[48] Request for Comments: 2598, Network Working Group, **V. Jacobson, K. Nichols** (Cisco Systems), **K. Poduri** (Bay Networks), June 1999, *An Expedited Forwarding PHB*

[49] IP QoS FAQ., http://www.QoSforum.com.

[50] **R. Braden,** *Integrated Services in the Internet Architecture*, Internet RFC 1633, 1994.

[51] **L. Zhang,** *RSVP: A New Resource ReSerVation Protocol*, *IEEE Networ*k, vol.**7**, pp. 8-18, September 1993.

[52] **Z. Zhang,** *Decoupling QoS Control from Core Routers: a Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services*, *In Proceedings of ACM SIGCOMM'0*0, Stockholm, Sweden, August 2000.

[53] **Octavian Pop 1, Tamás Máhr, Tímea Dreilinger, Róbert Szabó**, *Vendor-Independent Bandwidth Broker Architecture for DiffServ Networks*

[55] **F. Baker, C. Iturralde, F. le Faucher and B. Davie**, *Aggregation of RSVP for IPv4 and IPv6 Reservations*, Internet Draft, draft-ietf-issll-rsvp-aggr-02.txt, Mach 2000.

[56] **C. Centinkaya and E. Knightly** *Scalable Services via Egress Admission Control*, *In Proceedings of IEEE INFOCOM'0*0, Tel Aviv, Israel, March 2000.

[57] **J. Qiu and E. Knightly**, *Inter-class Resource Sharing using Statistical Service Envelopes*, *In Proceedings of IEEE INFOCOM'9*9, New York, NY, March 1999.

[58] **Octavian Pop 1, Tamás Máhr, Tímea Dreilinger, Róbert Szabó**, *Vendor-Independent Bandwidth Broker Architecture for DiffServ Networks*

[59] **Z. Zhang,** *Decoupling QoS Control from Core Routers: a Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services*, *In Proceedings of ACM SIGCOMM'0*0, Stockholm, Sweden, August 2000.

[60] **I. Stoica and H. Zhang**, *Providing Guaranteed Services without per Flow Management, In Proceedings of ACM SIGCOMM'*99, Cambridge, MA, August 1999.

[61] **F. Baker, C. Iturralde, F. le Faucher and B. Davie**, *Aggregation of RSVP for IPv4 and IPv6 Reservations*, Internet Draft, draft-ietf-issll-rsvp-aggr-02.txt, Mach 2000.

[63] **Fugui Wang, Prasant Mohapatra**, *Using differentiated services to support Internet telephony*, computer communications www.elsevier.com/locate/comcom

[64] **Liren Zhang, Li Zheng, Koh Soo Ngee**, *Effect of delay and delay jitter on voice/video over IP*, Computer Communications.

[65] **J.C. Bolot and A. V. Garcia**, *Control mechanisms for packet audio in the internet* in Proceedings of the Conference on Computer Communications (IEEE Infocom), (San Francisco, California), Mar.1996.

[66] **Perkins, C. and O. Hodson**, *Options for Repair of Streaming media*, RFC 2354, June 1998.

[67] **J. Rosenberg, H. Schulzrinne**, *An RTP Payload Format for Generic Forward Error Correction*, December 1999, RFC 2733, http://www.ietf.org/rfc/rfc2733.txt.

[68] **C.N Chuah, R.H.Katz**, *Network Provisioning and Resource management for ip telehony*, University of California, Berkeley, Report No.UCB /CSD–99-1061 ,September 1, 1999.

[69] **Y. Bernet**, *The Complementary Roles of RSVP and Differentiated Services in the Full-Service QoS Network, IEEE Communications Magazin*e, Vol. **38**, No. 2, February 2000.

[70] **M.Hamdaoui and P.Ramanathan**, *A dynamic priority assignment technique for streams with (m-k) firm deadlines*, IEEE Trans. Comput 44–(12) 1995, 1443-1451

[71] **M.F. Alam, M. Atiquzzaman, M.A.Karim,** *Traffic shaping for MPEG video transsmission over the next generation internet*

[72] Request for Comments: 2597, Network Working Group, **J. Heinanen** (Telia Finland), **F. Baker** (Cisco Systems), **W. Weiss** (Lucent Technologies), **J. Wroclawski** (MIT LCS), June 1999, *Assured Forwarding PHB Group*

[73] **Floyd S., and Jacobson V.**, *Random Early Detection gateways for Congestion Avoidance*, IEEE/ACM Transactions on Networking, Volume **1**, Number 4, August 1993, pp. 397-413.

[74] Request for Comments: 3140, Network Working Group, **D. Black, S. Brim, B. Carpenter, F. Le Faucheur**, Obsoletes: 2836 Category: Standards Track, June 2001 ,*Per Hop Behavior Identification Codes*

[75] Request for Comments: 3086, Network Working Group, **K. Nichols** (Packet Design), Category: Informational, **B. Carpenter** (IBM), April 2001, *Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification*

[76] Request for Comments: 3260, Network Working Group, **D. Grossman** (Motorola, Inc.), Updates: 2474, 2475, 2597, April 2002, Category: Informational, *New Terminology and Clarifications for Diffserv*

[77] Request for Comments: RFC 3198, A**. Westerinen, J. Schnizlein, J. Strassner**, **M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, S. Waldbusser**, November 2001 , *Terminology for Policy-Based Management*

[78] Request for Comments: RFC 3168, **K. Ramakrishnan, S. Floyd, D. Black**, September 2001, *The Addition of Explicit Congestion Notification (ECN) to IP*

[79] **Haiqing Chen, Hossam Hassanein, Hussein Mouftah**, *Providing Packet Loss Guarantees in Differentiated Services Architectures*

[80] **G. Zhang, H.T.Mouftah**, *End-to-End QoS Guarantees Over DiffServ Networks*, Department of Electrical and Computer Computer Engineering Queens University, Kingston, Ontario, Canada

[81] **A.Barberis, C.Casetti, J.C.De Martin, M.Meo**, *A simulation study of adaptive voice communications on IP networks*, Computer Communications 24 (2001), 757-767

[82] **Constantinos Dovrolis**, **Dimitrios Stiliadis**, and **Parameswaran Ramanathan**, *Proportional Differentiated Services: Delay Differentiation and Packet Scheduling*, 12 IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. **10**, NO.1, FEBRUARY 2002

[83] **C. Dovrolis and P. Ramanathan**, *A case for relative differentiated services and the proportional differentiation model IEEE Network*, vol. **13**, pp. 26–34, Sept./Oct. 1999.

[84] **Zesong Di, H.T.Mouftah**, Computer Communications, *QUIPS-II: a simulation tool for the design and performance evoluation of Diffserv-based networks*, http://www.elsevier.com/locate/comcom

[85] **Fasano Paolo**, CSELT, 10 June 1998, *IP QoS for voice –DiffServ Approach*, paolo.fasano@cselt.1t, presentation.

[86] **Jonathan Chao**, 2002 , *QoS Control in High Speed Networks*.

[87] **Bin Deng, Hao Liu, Minghua Zheng** , *Overview of QoS Routing***.**

BIOGRAPHY

Okan VURAL was born in Diyarbakır in 16 April of 1978 as the smallest boy of three brothers. Went to Şair Sırrı Hanım İlköğretim primary school and after that finished Diyarbakır Anatolian High School in Diyarbakır as the third student of the school in 1996. Then finished the Control and Computer Engineering of Istanbul Technical University in 2000 year as the tenth student of the department. Started to Computer Engineering graduate programme in 2000. Then, started to work in ADAM-ARGE as a software engineer. In September 2001 he started to work in ELİAR A.Ş. as a software engineer. Until that date he is working in there as a project engineer. In April 2003 he was a project manager in the same job.

He likes to play Turkish traditional musical enstrument *NEY*. Likes drawing portraits of human faces in different techiques.

Contact Information:

GSM : 0 535 293 43 82

Office : 0 262 644 31 92-93-94

E-Mail : vuralok@hotmail.com