

A traffic control system to manage bandwidth usage in IP networks supporting Differentiated Service

Myung-Sub Lee, Kwang-Jung Kim,
Chang-Hyeon Park

Department of Computer Engineering
Yeungnam University, #214-1, Dae-dong,
Kyongsan, Kyungbuk, 712-749, Korea

Joo-Hwan Oh

Department of Computer Engineering
Kyungwoon University, #55, Sandong-myeon,
Gumi, Kyungbuk, 730-852, Korea

Abstract— *This paper presents a traffic control agent that can perform the dynamic resource allocation by controlling traffic flows on a DiffServ network. In addition, this paper presents a router that can support DiffServ on Linux to support selective QoS in IP network environment. To implement a method for selective traffic transmission based on priority on a DiffServ router, this paper changes the queuing discipline of Linux, and presents the traffic control agent so that it can efficiently control routers, efficiently allocates network resources according to service requests, and relocate resources in response to state changes of the network. Particularly for the efficient processing of Assured Forwarding(AF) Per Hop Behavior(PHB), this paper proposes an ACWF2Q+ packet scheduler on a DiffServ router to enhance the throughput of packet transmission and the fairness of traffic services.*

Keywords : QoS, PHB, DiffServ, ACWF2Q+

I. INTRODUCTION

For the last few years, IETF(Internet Engineering Task Force) WG(Working Group) have carried out research on IP(Internet Protocol) packet transfer mode based on new service models to support QoS that requires real-time application services, and representative services are IntServ(Integrated Services)[1] and DiffServ(Differentiated Services)[2]. IntServ model transmits packets after reserving resources for each flow using RSVP (Resource reSerVation Protocol) to overcome delays in end-to-end packet transmission.

However, to support IntServe model, each intermediate router has to store information about flow, so it needs a storage space and may have high process overhead. Particularly in Internet backbone routers, the transmission rate is very high and the number of linked flows is very large, so it is hard to support IntServe model at core nodes[3-5].

Therefore, the structure of DiffServ model and relevant standards are being developed as a new service model that may overcome the limitations of IntServ model, which has a scalability problem, and be applied to Internet backbone networks. First of all, DiffServ model purposes to classify

QoS into a number of classes and guarantee the transmission quality of service according to each class.

Accordingly, DiffServ model defines differentiated services by marking QoS information on specific fields of IP headers(IPv4: ToS Field, IPv6: Traffic Class Field), and appropriate transmission(PHB: per hop behavior)[6] is carried out according to the QoS information.

DiffServ model tries to solve the scalability problem of IntServ structure by dividing the role of boundary nodes from that of core nodes. DiffServ needs a device to determine resource allocation based on the resources required by clients and SLA(Service Level Agreement)[7] information set in network nodes, and a device with such a function is called BB(Bandwidth Broker)[8].

DiffServ model, which was proposed to overcome problems in IntServ, is under development for its implementation. However, the proposed model has not been deployed in networks yet. The reasons are discussed through analyzing results from previous research on proposed DiffServ structure[9-11].

First, while the structure of IntServ model guarantees QoS of each flow that of DiffServ cannot support QoS of each flow and its QoS is not perfect because it distinguishes services to be guaranteed by class. Thus, although DiffServ model solves the scalability problem in the core network so is superior to IntServ model in application to actual networks, it cannot provides perfect QoS because it does not support the function to guarantee QoS for each flow.

Second, DiffServ models proposed so far does not mention the use of a signal protocol. That is, because DiffServ model adds class information concerning QoS to ToS field of IP headers and carries out each PHB process based on the information, the necessity of a separate signal protocol has not been raised. However, to support dynamic SLA in DiffServ model, the use of a signal protocol is essential for communication between BB and boundary routers within a differentiated service network, between hosts and boundary routers within a differentiated service network, and between BB even if it may be necessary for resource reservation.

Third, the role of BB is very important for dynamic SLA interference. However, IETF DiffServ Specification does not

clearly define the role of BB. Therefore, it is necessary to define the role and function of BB for DiffServ.

To solve problems as mentioned above, this paper proposes a traffic control agent that performs dynamic resource allocation in response to demand for bandwidth from routers and clients that support DiffServ, which guarantees QoS selectively in IP network environment.

The traffic control agent proposed in this paper carries out the function of dynamic resource allocation by collecting and analyzing information about traffic on network nodes, and for dynamic resource allocation, defines a signaling system between the traffic control agent and clients, between the traffic control agent and routers, and between the traffic control agent and another traffic control agent.

In addition, for DiffServ-supporting routers, ACWF2Q+(Asynchronous and Class based WF2Q+) packet scheduler, which provides the functions of asynchronous queuing and by-class queuing, is proposed to improve the processing rate and fairness of AF PHB and guarantee QoS for each explicit forwarding(EF) micro flow.

II. TRAFFIC CONTROL AGENT

The traffic control agent in this paper performs dynamic resource management based on resources required by clients and SLA(Service Level Agreement) information. To perform dynamic resource management, a traffic control agent has a database to store bandwidth allocated to each SLA and uses it as basic data to decide resource allocation in the future.

If a client requests a service to a boundary router of the network, the router sends the service request information to the corresponding traffic control agent based on SLA information including service type, target transmission rate, the maximum burst size and service time. On receiving service request information, the traffic control agent checks if it can allocate the requested bandwidth and, if available, sends admission information to the boundary router.

That is, a traffic control agent manages the whole resources in the differentiated service network, and decides admission/rejection according to the policy of differentiated service network. By doing so, it performs the functions of communication between internal routers as well as negotiation with neighboring differentiated service networks.

To perform these functions, the traffic control agent in this paper has a two-tier structure.

The first tier is for inner domain resource management that handles resource allocation inside the differentiated service, managing resources through communication between the traffic control agent and routers inside the domain.

The second tier is for inter domain resource management that supplies and allocates resources on the network boundary between two domains. For this, it exchanges

information such as the traffic volume and the traffic type between two domains through admission control between traffic control agents.

As shown in Fig. 1, the traffic control agent proposed in this paper is composed of an admission control module, a resource management module and a congestion analysis module. These modules are explained in following sections.

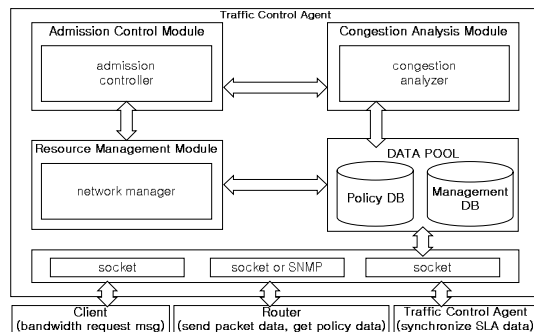


Fig. 1 Structured diagram of traffic control agents for flow management

2.1 Admission Control Module

The admission control module decides whether a new request will be admitted or not so that the sum of reservation rate of flows on the link does not exceed the capacity of the link. This module can perform SLA negotiation statically referring to policy database or dynamically in connection with the resource management module.

The initial policy of this module is set by the network administrator and the defined policy is stored into Policy DB. The admission controller decides the admission of service after performing SLA referring to Policy DB in response to a client's request. If the request is admitted, the traffic control agent performs traffic control through communication with a flow manager in the DiffServ router.

The flow manager performs functions such as dynamic filter generation, queue management policy definition and queue weight allocation. Dynamic filter generation creates "u32" filter in Ingress interface of a boundary router to meter packets, and performs DSCP marking of traffic classified through "tc_index" filter in Egress interface.

Queue management policy and weight setting for allocating reserved resources is applied equally to boundary routers and core routers.

2.2 Resource Management Module

To process SLA information requested by clients, the resource management module collects information about network resources from routers and calculates the availability of service. The information collected from routers is stored in Management DB, and information from calculating the availability of service is stored in Policy DB. Information stored in Policy DB is used by the admission control module to decide the admission of clients' requests.

In addition, the network utilization rate in a differentiation service domain is calculated and used as information for SLA between clients and a traffic control agent.

As shown in Fig. 2, the resource management module in this paper uses a packet acquisition method and SNMP module. The packet acquisition method is used to check the traffic transmitted to the differentiated service network by service unit, and is implemented using LibPcap library.

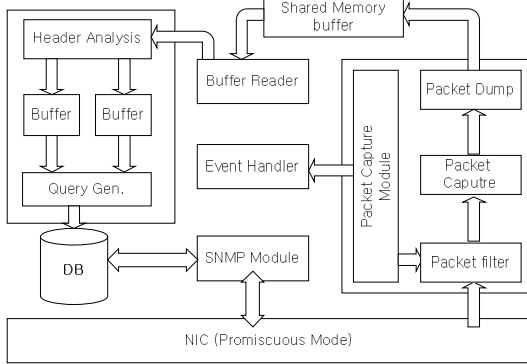


Fig. 2 Structured diagram of Resource Management Module

In this module, packet collection and database storage are executed as separate process, and a shared memory technique is used for data sharing among processes. The packet acquisition method acquires packets through packet filtering process in a network device, decodes the headers according to some routines and stores them into the shared memory buffer. At that time, the data conversion process reads the data, converts them into structure fit for database tables through a series of conversion processes, and store them.

2.3 Congestion Analysis Module

The congestion analysis module analyzes transmission rate, utilization rate and error rate for each packet flow based on management data collected by the resource management module and uses them as basic material for. Equation (1) through (5) is used by this module to analyze network utilization rate and error rate. Variables in these equations are summarized in Table 1.

Table 1. Variables used in Equation (1) through (5)

Variable	Explanation
x	Previous polling time at polling cycle
t	Polling cycle
$ifInOctets$	Total number of octets received at the interface
$ifOutOctets$	Total number of octets sent outside the interface
$sysUpTime$	Time from the last re-initialization of the system
$ifSpeed$	Current bandwidth of the interface (bit per second)
$ifInError$	Number of packets not delivered to upper layers due to error
$totalPktIn$	Total number of packets arrived

First, to estimate the network utilization rate of FDDI network, measure the input/output traffic volume of all routers in FDDI network and divide the sum into two. To get the input . output traffic volume of all routers, sum the total number of bits of packets sent by the sender and the total number of bits received by the receiver and divide it into the total bandwidth of the network link as Equation (1).

Input/Output traffic of router :

$$(total_bit_sent + total_bits_received) / bandwidth \quad (1)$$

Because the maximum transmission rate of FDDI network is 100Mbps, the utilization rate can be estimated as Equation (2).

FDDI Utilization :

$$\frac{1}{2} \frac{[ifInOctets_{(x+t)} - ifInOctets_{(x)} + ifOutOctets_{(x+t)} - ifOutOctets_{(x)} * 8]}{(sysUpTime_{(x+t)} - sysUpTime_{(x)}) * ifSpeed * 100} \quad (2)$$

In general, if the utilization rate of FDDI network exceeds 90%, it is regarded as overload. It is not serious for the utilization rate to go over 90% temporarily, but if the average utilization rate is over 90%, the entire network is overloaded, so it must be reconstructed or upgraded. To compute network traffic utilization rate, Congestion Control Module uses different analysis equations according to component network. Basically Ethernet uses the broadcasting transmission mode.

Thus, to measure the utilization rate of Ethernet networks, sum all input . output traffic and divide the sum of traffic by the maximum bandwidth. Equation (3) shows the equation for analyzing Ethernet traffic utilization rate.

Ethernet Utilization :

$$\frac{[ifInOctets_{(x+t)} - ifInOctets_{(x)} + ifOutOctets_{(x+t)} - ifOutOctets_{(x)} * 8]}{(sysUpTime_{(x+t)} - sysUpTime_{(x)}) * ifSpeed * 100} \quad (3)$$

For serial link utilization rate, there are two transmission modes, which are half duplex mode and full-duplex mode. In case of half-duplex serial link, the sum of input traffic becomes the utilization rate of the link, and in case of full-duplex serial link, there are two lines according to the direction of link connection. In serial links, if the utilization rate is over 90%, the network is regarded as overloaded. Like FDDI network, it is not serious for the utilization rate to go over 90% temporarily, but if the average utilization rate is over 90%, the network is considered to have a serious problem, so it is necessary to reconstruct the network or upgrade equipment.

Full-duplex serial link utilization :

Max[(total_bit_sent + total_bit_received)/bandwidth] :

$$\text{Max} \left[\frac{[ifInOctets_{(x+t)} - ifInOctets_{(x)}], (ifOutOctets_{(x+t)} - ifOutOctets_{(x)}) * 8]}{(sysUpTime_{[x+t,t]} - sysUpTime_{(x)}) * ifSpeed * 100} \right] \quad (4)$$

Equation (4) shows equations to analyze the utilization rate of full-duplex serial link for each direction of link connection. As for error rate, if over 1% of the entire bandwidth is proved to be error, the network administrator must check network devices. Equation (5) shows the error rate analysis equation.

Traffic error rate :

$$\left[\frac{\text{iflnError}_{(x+t)} - \text{iflnError}_{(x)}}{\text{totalPktIn}_{(x+t)} - \text{totalPktIn}_{(x)}} \right] \quad (5)$$

In Equation (5) means the total number of input packets. As Equation (6), the total number of input packets is the sum of the number of input unicast packets, the number of input broadcast packets, and the number of input multicast packets.

Tatal input packet count :

$$\text{totalPktIn} = (\text{iflnUcastPkts} + \text{iflnBroadcastPkts} + \text{iflnMulticastPkts}) \quad (6)$$

III. DIFFSERV ROUTER

DiffServ router performs functions such as policing and marking in the boundary router inside DiffServ domain, and the core router executes PHB using the first 6 bits of ToS field in the IP header. Because DiffServ operates based on groups, it treats flows not individually but as a group, namely, a single flow into the differentiated service network. In addition, when packets are sent and received from other differentiated service domains, all boundary routers must set a queue management policy fit for their domains to packets arrived from other domains. Table 2 shows parameters to compose the dsmark queue management policy of boundary routers.

Table 2. Parameters composing dsmark

Variable name / tc keyword	Value	Default
indices	2^n	none
default_index	0 ... indices-1	absent
set_tc_index	none(flag)	absent
mask	0 ... 0xff	0xff
value	0 ... 0xff	0

In Table 2, parameters composing dsmark are indices, default_index and set_tc_index. indices is expressed as a pair of (mark, value) indicating the maximum table size. When setting indices, the maximum table size is calculated and set and the value is $2n$.

set_tc_index informs dsmark of the value of DS field, and store it into the filter. Fig. 3 represents the operation procedure of dsmark queue management policy.

The procedure is described as follows.

- ① If 'set_tc_index' is set in the input packet, DS field checks it and stores filtered value into 'skb->tc_index'.
- ② Classifier is executed and returns class ID stored in 'skb->tc_index'. At that time, if a filter is not found, 'default_index' option is applied. If all these are not satisfactory, the value of 'tc_index' filter is reset.
- ③ After ① and ② are completed, the packet moves to 'inner qdisc'. Here, the packet reuses the return value in the filter (class ID returned after being stored in skb->tc_index), and the value is used as indices for table (mask, value).
- ④ Finally, the value of DS field allocated to the packet is converted by Equation (7).

$$\text{new_DS_field} = (\text{old_DS_field} \& \text{mask}) \mid \text{value} \quad (7)$$

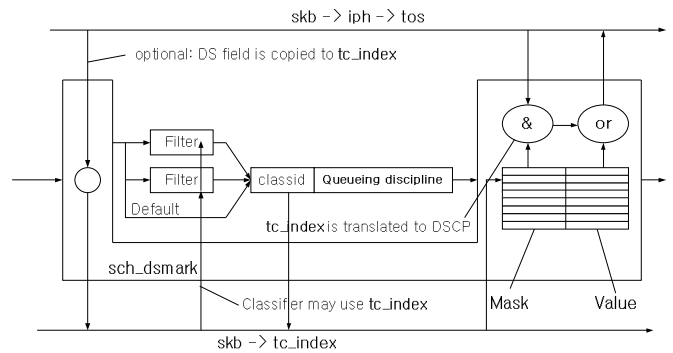


Fig. 3 dsmark queue management policy

Equation 7 is to abstract a new DSCP value to execute PHB of the core router at tc_index of the boundary router. tc_index filter executes masking using 0xfc in Ds field abstracted from input traffic, and the masked bit shifts rightward two bits to convert a new DSCP value.

In core router, each PHB is classified (BA classification) based on the DSCP value of packets classified in the core router inside differentiated service network and entered into its queue.

To provide quality service to packets set as EF, AF and default PHB, appropriate buffer management policy and scheduling are executed. While the setting of a boundary router is expensive and dynamic, the setting of core router is cheap and static.

In the core router, packets are classified by the filter into AF, EF and BE services, and classified packets are scheduled according to the operation of a packet scheduler connected to the corresponding class.

Fig. 4 shows the detailed structure of a core router. In the figure, ds_mark queue management policy is directly related to the output device, and here ToS bytes are abstracted from packets entered and transmitted to tc_index filter.

tc_index filter executes masking by applying 0xfc to ToS bytes abstracted, and abstracts DSCP value by shifting the masked bit rightward by two bits. Abstracted DSCP value is stored in skb->tc_index and packets are transferred to CBQ (Class Based Queue).

IV. PERFORMANCE ANALYSIS

This section analyzes the performance of the traffic control system in DiffServ environment proposed in this paper through a real test. The real test model for the proposed model is as shown in Figure 6. It measures system performance using delay, jitter, and packet drop rate. How to measure delay is defined in RFC 2679[14]. Delay is measured using the difference between the wire time of the last bit of a packet recorded by the receiver side and the wire time of the packet arrived at the link recorded by the sender side. Jitter is measured using the method defined in IPPM WG[15][16] of IETF as well as delay, and indicates the continuity of two packets. Thus, two packets are necessary to measure jitter. If the delay of i th packet is D_i , its jitter becomes $|D_i - D_{i-1}|$.

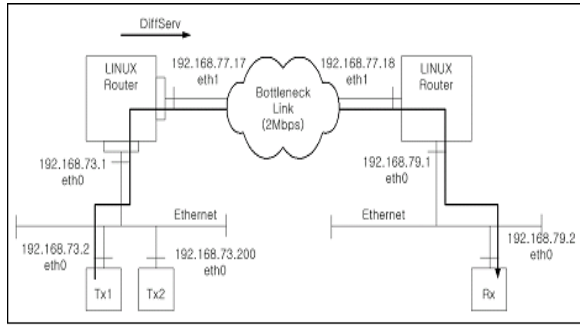


Fig. 6. Real test evaluation model

4.1 AF(Assured Forwarding) PHB

Test 1 measured traffic processing rate and service fairness when applying srTCM, RED buffer management policy and PQ (Priority Queue) to three TCP flows and one UDP flow and when applying trTCM, WRED and ACWF²Q⁺ scheduling algorithm proposed in this paper to the same flows.

To calculate the fairness of AF PHB, this paper used Equation 8[17].

$$FI = \frac{(\sum_i x_i)^2}{N \times \sum_i x_i^2} \quad (8)$$

In Equation 8, FI (Fairness Index) is to value fairness, which ranges between 0 and 1. X_i indicates the average processing rate of the i th traffic source, and N means the total number of traffics in the sender. If FI is closer to 1, it means that the distribution of bandwidth through which traffics are delivered to the receiver is fairer.

Figure 7 is the result of testing average processing rates when applying srTCM, RED buffer management policy and PQ (Priority Queue) and when applying trTCM, WRED and ACWF²Q⁺ scheduling algorithm, which are traffic control conditions proposed in this paper. First, according to the result of applying srTCM, RED and PQ, each flow before bottleneck causes traffic congestion satisfies

CIR(Committed Information Rate) and is fairly allocated surplus bandwidth. As shown in Figure 7, however, after congestion has happened, UDP micro-flow is allocated not only its own bandwidth but also surplus bandwidth and most of TCP flows are discarded and not allocated bandwidth at all.

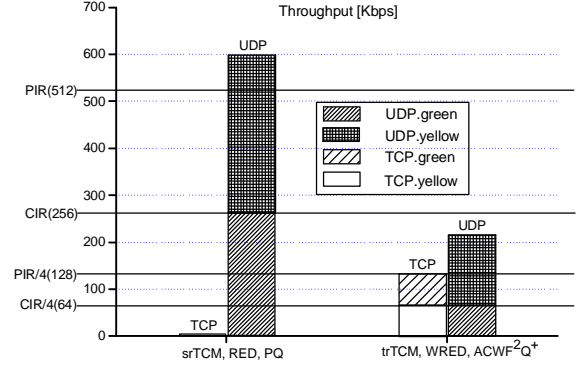


Fig. 7 Average processing rate of TCP and UDP micro-flows

Second, according to the result of applying trTCM, WRED, ACWF²Q⁺, which are traffic control conditions proposed in this paper, both TCP and UDP flows are fairly allocated traffics for AF traffics as well as for surplus bandwidth as shown in Figure 8. In addition, fairness is improved if CIR is set closer to PIF (Peak Information Rate) in trTCM setting. The bigger the difference between PIF and CIR, the more of available bandwidth UDP micro-flow occupies.

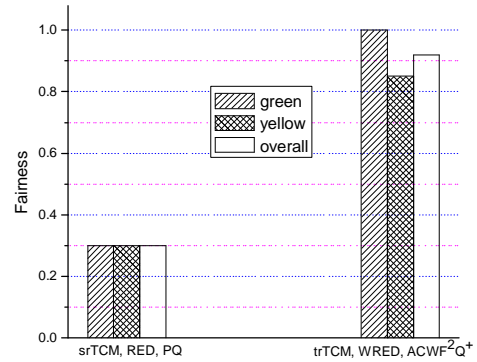


Fig. 8. Fairness of TCP and UDP micro-flow

4.2 EF(Expected Forwarding) PHB

This section perform a test on what influence the aggregation of EF traffic loads and EF micro-flows has on the transmission rate of the whole EF traffics. Performance factors used in measuring transmission rate were drop rate, delay and jitter.

This test proved that packet dropping in EF class is related to the number of flows composed of classes. In

addition, it showed that even if the size of EF packet increases, the packet drop rate is as low as 1.2% because the rate of policing by PQ policier is applied differently according to packet size. It was also found that the increase of EF traffic load increases the packet drop rate but decreases delay. Jitter increases along with the number of flows because of EF traffic load, and then remains constant since the number of EF micro-flows reaches 12.

Figure 9 shows packet drop rates when a number of EF micro-flows are aggregated. In the figure, the drop rate of EF packets increases with the increase of EF micro-flows in the class. As shown in Figure 11, EF traffic load begins to occur from 600Kpbs, which is ten thirds of the whole link capacity and packet dropping begins to occur when the number of EF micro-flows reaches 16.

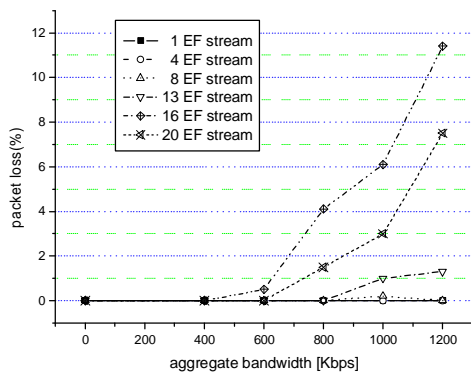


Fig. 9 Packet drop rate according to the number of EF micro-flows (pkt size: 256byte)

In addition, the increase of the number of EF micro-flows reduces EF load caused by packet dropping. Figure 10 shows that packet dropping occurs when the size of EF frame increases from 256 bytes to 1024 bytes and the maximum packet drop rate is less than 1.2%.

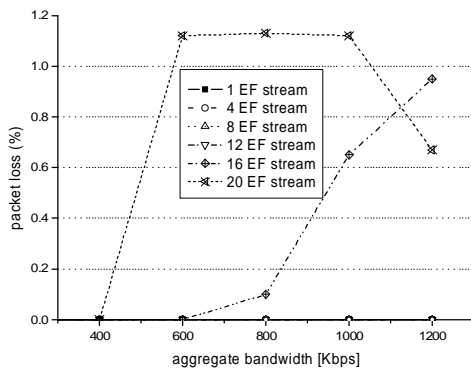


Fig. 10 Packet drop rate according to the size of EF frame(pkt size: 1024byte)

V. CONCLUSION

This paper proposes a traffic control agent that can perform the dynamic resource allocation by controlling traffic flows on a DiffServ network. In addition, this paper proposes a router that can support DiffServ on Linux to support selective QoS in IP network environment.

To implement the router supporting DiffServ, this paper proposes a conceptual model of a DiffServ router as well as a mathematical model of traffic flow between edge routers.

To implement a method for selective traffic transmission based on priority on a DiffServ router in Linux, and presents the traffic control agent so that it can efficiently control routers, efficiently allocates network resources according to service requests, and relocate resources in response to state changes of the network.

Particularly for the efficient processing of Assured Forwarding(AF) Per Hop Behavior(PHB), this thesis proposes an ACWF²Q⁺ packet scheduler on a DiffServ router to enhance the throughput of packet transmission and the fairness of traffic services. The ACWF²Q⁺ packet scheduler enhances the throughput by solving the problem of restricting the use of excess bandwidth in AF PHB, and enhances fairness by solving the problem that, some packets of low priority are not provided with services at all, in worst case in the existing priority-based queuing.

REFERENCES

- [1] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," *IETF RFC 1633*, June 1994.
- [2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," *IETF RFC 2475*, December 1998.
- [3] W. Almesberger, "Linux Traffic Control Implementation Overview," *Technical Report, EPFL*, November 1998.
- [4] P. Salambier, S. Shenker, C. Partridge, R. Guerin, "Specification of Guaranteed Quality of Service," *IETF RFC 2212*, September 1997.
- [5] A. Parekh, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks," *Ph.D. dissertation, MIT*, February 1992.
- [6] K. Nicholas, S. Blake, F. Barker and D. Blake, "Definition of the Differentiated Services Field(DS Field) in the IPv4 and IPv6 Headers," *IETF RFC 2474*, December 1998.
- [7] D. Verma, "Supporting Service Level Agreement in IP Networks," *Macmillan Technical Publishing*, 1999..
- [8] B. Evans, "Differentiated Service Implementation," <http://www.itc.ku.edu/~kdrao/BB/>.
- [9] Andreas Terzis et al., "A Prototype Implementation of the Two-Tier Architecture for Differentiated Services," *RTA 99*.
- [10] Benjamin Teitelbaum and et al., "Internet2 QBone : A test Bed for Differentiated Services", *INET 99*.
- [11] Hyogon-Kim, "Architectural and Practical Problems in DiffServ Deployment," The third next-generation internet workshop : *Session III-QoS*, November 2000.
- [12] J. C. R. Bennett, H. Zhang, "Why WFQ Is Not Good Enough for Integrated Services Networks," *Proceedings of NOSSDAV'96*, Zushi, Japan, April 1996.
- [13] J. C. R. Bennett, H. Zhang, "WF²Q": Worst-case fair weighted fair queueing," *Proceedings of IEEE INFOCOM'96*, San Francisco, CA, pp. 120-128, March 1996.