Using Active Networks Technology for Dynamic QoS

T. Tansupasiri [a,*] , K. Kanchanasut [a] , C. Barakat [b] , P. Jacquet [c]

[a] *Asian Institute of Technology, Computer Science and Information Management Program, P.O. Box 4, Pathumthani, 12120, Thailand*

[b] *INRIA, PLANETE research group, 2004, route des Lucioles, 06902 Sophia Antipolis, France*

[c] *INRIA, HIPERCOM research group, B.P. 105, Rocquencourt, 78153 Le Chesnay Cedex, France*

Abstract

We propose a dynamic QoS, or D-QoS, model where QoS settings can be automatically reconfigured based upon requests from authorized users. Different levels of privilege can be assigned to users enabling higher privileged users to interrupt the network flows belonging to those of lower privileged levels. To request for a special QoS treatment, a user can issue an active packet to interrupt any active node along its flow path which is D-QoS enabled. The request for a specific interruption level is approved by a D-QoS enabled node which allows for multi-level interruptions to be handled. After an interrupting flow has completed transmitting all its packets, D-QoS enabled node can resume its services for those pending flows which are of lower privilege levels. In this paper, we describe the overall concept of D-QoS and demonstrate how it can be implemented by a small prototype. Using simulation, we show that the proposed system can provide assurance for privileged flows with an improved network utilization where bandwidth is shared among the flows according to the levels of privilege. D-QoS should be deployed on those bottleneck hops with limited bandwidth on the edge network to ensure the best service is given to privileged users.

Key words: QoS, Active Network, Bandwidth Management

* Corresponding author. Tel./Fax +66-2524-6619

Email addresses: fon@cs.ait.ac.th (T. Tansupasiri), kk@cs.ait.ac.th (K. Kanchanasut), Chadi.Barakat@sophia.inria.fr (C. Barakat), Philippe.Jacquet@inria.fr (P. Jacquet).

1 Introduction

The concept of active networks was introduced in 1994 by Defense Advanced Research Projects (DARPA) research community, as a future direction of networking system [1]. In addition to packet forwarding mechanism in traditional IP network, active networks allow the network nodes to perform user or application specific computations on user data passing through them. In active networks, normal packets are replaced with active packets containing small programs and possibly data. Network nodes are substituted by active nodes capable of performing computations on packets as requested by the applications. Thus, the role of the network is changed from a passive carrier of data into a general computation engine. New network services can, therefore, be faster innovated and deployed. These network services are, for example, IP multicast, mobile IP and web caching, as they can be effectively provided in the network layer.

One of the services that could also be offered based on active network is the quality of services, QoS. As Internet traffic can be generated possibly from anywhere and at any time, it is desirable that QoS be dynamically adaptable to user requirements thus tailoring to the demand of the Internet users. Active network technology can be used as a useful tool for providing dynamic or on demand QoS where active nodes can adjust their QoS configuration as instructed by the programs in active packets.

Integrated Service (IntServ) model described in RFC 1633 [2] provides resource sharing mechanism which could be requested dynamically per-flow. The model deploys resource reservation mechanism in which network resources are reserved along each communication path. However, this model suffers from its scalability problem due to the overhead on maintaining the path state information at each node or router. Several attempts to provide scalable dynamic QoS have been made as in [3–6]. They combine IntServ with another QoS model, Differentiated Service (DiffServ) [7] which is known to be scalable. Rather than providing per flow QoS, DiffServ provides coarse grained QoS for traffic aggregates or classes and applies different per-hop services to different classes. These works share the idea on deploying IntServ at the edge and DiffServ at the core networks. They introduce the use of a signaling mechanism in IntServ to communicate the QoS requirements to DiffServ. An additional network element called bandwidth broker, is introduced to monitor the resources usage within the domain and determines the admission control dynamically. Although this idea allows dynamic admission control, but flows are assign to a static preconfiguration in [3–5]. An extended idea for dynamic resource allocation is explained in [6]. The bandwidth broker, in this case, is also responsible for monitoring the bandwidth usage of the client and the state of the network. As the actual traffic rarely approaches the reserved bandwidth

for the peak rate, a portion of the unused bandwidth can be reclaimed to the pool of available bandwidth and reallocated to other flows. However, these works require that some elements in DiffServ network are IntServ-aware for such interoperation and restrict the resource allocation with DiffServ preconfiguration. Hence, they cannot really provide QoS on demand which could be requested directly by authorized users.

In [8,9], dynamic QoS has been proposed where active network technology is deployed. A QoS scheme particularly for MPEG video transmission is proposed in [8] with an active buffer management and active packet scheduling mechanisms. Packets are classified into queues according to their contents. Packets containing self-contained video frames receive better service at network nodes while less important packets are preferably dropped. In [9], an integration of DiffServ, active network and policy based management has been proposed to automate the configuration at DiffServ nodes. The proposed model, namely DiffServ Active Control Architecture (DACA), consists of DiffServ active components that allow flexible control over the classifier, meter, dropper and scheduler algorithms through policies. It provides a repository of the active codes for those mechanisms that could be loaded on active nodes on demand and a Dynamic Management Information Base (DMIB) where the managed objects and information can be changed or added dynamically. However, this work focuses on dynamic QoS within the DiffServ framework only.

In [10], Tansupasiri and Kanchanasut proposed another concept of using active networks to provide dynamic QoS. The proposed system, namely Dynamic Quality of Service (D-QoS), allows the QoS requirements to be reconfigured dynamically on network nodes upon receiving the requests within the active packets from authorized users. An authorized user can request for an interruption of a privileged flow transmission, in the same manner as the interruption of a super user process in the operating system where system resources are first given to the super user. An example of the privileged flows is the time sensitive flow that requires high precision of data transmission like telesurgery flow.

Each D-QoS node is triggered and the normal operation mode is automatically changed to the interruption mode upon receiving a request from an authorized user. In the interruption mode, the privileged flow is then transmitted with the privilege over other existing flows. Even though QoS can be characterized by throughput, delay, jitter and/or loss, the model relies on a narrow QoS concept based on the bandwidth sharing among flows. The system also allows multi-level interruptions where privileged flows with different priority levels can coexist.

Dynamic QoS can be applied regardless of existing network settings provided that some nodes on the network are D-QoS enabled. The existing network

could be a best-effort network without any QoS or it can be a network which deploys some other QoS models, such as DiffServ. In this particular study, we have chosen to assume that existing network is employing DiffServ since DiffServ is a widely used and accepted QoS model.

In [10], the D-QoS model has been successfully implemented and tested through the prototype and experiments, where the model has operated satisfactorily in both normal operation mode with DiffServ and interruption mode. In the normal operation mode where DiffServ is the only QoS model, D-QoS follows the service level agreement (SLA) of DiffServ, but once in the interruption mode, the SLA no longer applies.

D-QoS nodes are designed such that they could co-exist with other QoS models, thus the scheme could be deployed incrementally. We can start by deploying the scheme in congested routers and then extend it to other routers where D-QoS nodes may substitute non-active nodes afterwards to ensure the quality of the privileged flow transmissions. A network administrator may place D-QoS nodes only on bottleneck hops of the edge network with tight bandwidth. The system can be deployed without tunneling effort as needed for the Active Network Backbone (ABone) [11], a dynamic virtual network or Internet overlay system, where isolated active nodes are connected through tunnels with IP encapsulation. We mainly consider deploying D-QoS within an edge network under same administrator such as within the same autonomous system (AS). Hence, the security issue for user authentication is not as prominent as in a large scale network and is outside the scope of this paper.

This paper describes how D-QoS system automatically adjusts its QoS settings according to user requests. The prototype of the D-QoS model is explained together with the experiments to demonstrate the realization of the model. An extensive work to that reported in [10], is presented to study the behavior of the proposed conceptual model through simulation with Internet traffic patterns and provide analytical study of the model. The paper is organized as follows. Details of the D-QoS system is presented in Section 2. A summary of the prototype implementation and the experiments is provided in Section 3. In Section 4, we present the study of the system performance where the simulations with ns-2 [12] are reported in Section 4.1. We provide an analytical investigation on delays using delay probability generating function in Section 4.2. The security and scalability characteristics are discussed in Section 4.3. Section 5 investigate other works related to dynamic QoS and, finally, we conclude the paper in Section 6.

2 Dynamic Quality of Service (D-QoS) System

Our proposed model allows bandwidth allocation be dynamically adjusted based on the concept of active network. An authorized user can send an active packet requesting for a QoS reconfiguration on D-QoS nodes whenever it has a need for a prioritized flow to get through. D-QoS system allows the authorized users to perform interruptions on the nodes such that their privileged flows can be transmitted through the network with the priority on the expense of other existing flows. The interruption mechanism of the D-QoS model follows the concept used in the interruption of a super user process in the operating system, where the resources are first given to the super user process while other processes are suspended. In D-QoS, a number of interruption levels are provided allowing a set of privileged flows to be transmitted according to their levels of privilege.

The model relies on the concept of active IP network presented in [13,14] where active IP nodes can coexist and interoperate with normal IP nodes. Each active packet, called capsule, carries both the program fragment and data parts. The program part is stored within packet options that are recognized only on those active nodes and triggers the interruption to be performed on the nodes. Normal IP nodes ignore the program part and treat these capsules as normal IP packets.

A flow is a sequence of packets and can be uniquely identified by the information placed in packet headers, which are source and destination IP addresses, source and destination port number and protocol. We refer to flows belonging to those authorized users as privileged flows. A privileged flow transmission can be sent as a stream of capsules containing the same program part requesting for an interruption. These capsules are forwarded according to any routing mechanism in used. When a D-QoS node receives a capsule, an interruption is generated for the flow it belongs to. When an alternate path has been chosen by dynamic routing mechanism, the interruption generated on D-QoS nodes on the previous path are removed on timeout where the nodes turn to normal state. This allows D-QoS to co-operate with any routing mechanisms, both static and dynamic, where the flow is guaranteed to be transmitted with the privilege over any D-QoS nodes along its flow path.

D-QoS can be applied regardless of existing network settings. The network could be a best-effort network without any QoS or it can be a network which deploys some other QoS models, such as DiffServ. In our demonstration, we assume that the network is running with DiffServ since it is widely adopted on IP networks due to its simplicity and scalability. Hence it is expected that in the practical environment, we would be likely to operate in an environment with DiffServ is present.

In the normal operation mode in our prototypical implementation of D-QoS, the system offers two main classes in DiffServ, Expedited Forwarding (EF) and Assured Forwarding (AF) classes following the specification in RFC 3246 [15] and RFC 2597 [16] respectively.

The bandwidth sharing between classes is predefined and preconfigured where EF is designed for traffic with low loss, low delay and low jitter requirements, and AF offers relative bandwidth sharing and drop characteristics among the aggregates. However, the choice of the QoS model for the normal operation mode is not limited to DiffServ, any other models can also be provided, even the best effort service.

To transmit a privileged flow, an authorized user can send active packets or capsules, each of which contains the interruption request in its program part and flow payload in its data part. When a D-QoS node receives any of these capsules, it is triggered to change its operation mode into the interruption mode where its QoS setting is reconfigured according to the request. The capsule is then forwarded towards the destination of the flow to generate the interruption on any D-QoS nodes along the communication path.

In this interruption mode, the D-QoS node no longer supports DiffServ, but allows the privileged flow to be transmitted with the highest possible bandwidth at the expense of other lower priority traffic. Whenever the node receives a packet belonging to the privileged flow, the packet is always transmitted before packets of non-privileged flows. However, the suspended flows can also be serviced in the absence of the privileged flow.

The queueing mechanism used within this interruption mode aims to give higher priority to the privileged flow. Other flows previously classified into DiffServ's EF and AF classes are transmitted with the lowest priority. To provide different services for time-sensitive (EF) flows and non-time sensitive (AF) flows, the two lowest priority levels are reserved for each one of these two flow types. The time-sensitive flows previously classified into DiffServ's EF class are placed in the second lowest priority level, while other flows previously belonging to DiffServ's AF classes are given the lowest priority level. Thus, EF traffic receives lower-loss, lower-delay and lower-jitter comparing to AF traffic.

D-QoS system provides a fixed range of interruption levels, each of which can be assigned to a particular privileged flow. These levels determine the priorities among the privileged flows. The queueing structure in the interruption mode is illustrated in Fig. 1, with the two lowest priority queues reserved as mentioned above. Thus, the first interruption request received would change a D-QoS node operating in its normal operation mode to the interruption mode with three output queues of different priorities where the highest priority is for the

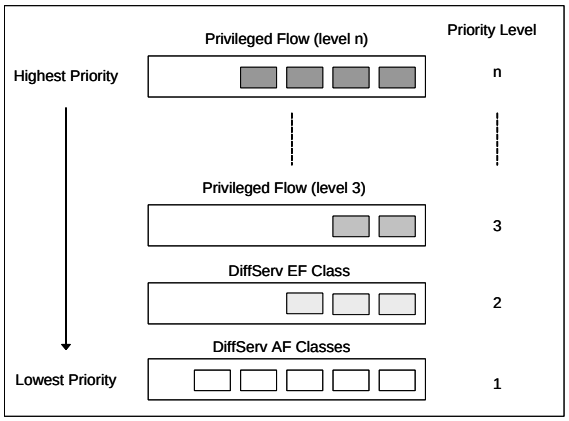

Fig. 1. Queueing Architecture for Interruption Mode on D-QoS Node

privileged flow.

An authorized user may request for a specific interruption level according to the flow requirement. The requested level is examined by each D-QoS enabled node for its availability. If the privileged flow is requesting for an interruption level which is occupied by another privileged flow, an interruption with a lower level is generated instead. An interruption request results in an insertion of a new queue with the requested priority level into the output queueing structure. At the end of the privileged flow transmission, the queue is removed from the system. Thus, the number of queues shown in Fig. 1 may vary according to the number of active privileged flows. Once all the privileged flows have completed their transmissions, the system resumes its normal operation on DiffServ.

3 D-QoS Prototype and Experiments

A prototype implementation of D-QoS has been constructed to demonstrate the concept with three D-QoS nodes, as shown in Fig. 2, where each D-QoS node is a PC router running FreeBSD. Each D-QoS node is implemented as an active node by emulating an active IP node that provides restricted primitives related to the interruption handling. The program code transmission is separated from the data transmission in this particular implementation. Program code for interruption request or removal is sent to D-QoS nodes through an opened UDP socket, while flow payload is transmitted as normal IP packets. Prior to the flow transmission, the request is sent to all D-QoS nodes, one by one. A request contains the information used to identify a particular flow and the required interruption level as the parameters for the primitive. A specific flow is identified by the source and destination addresses, source and destination port numbers and protocol identification. Upon receiving an active packet, its program part is examined and the appropriate action is performed on the D-QoS node, as instructed by the program. Once the D-QoS

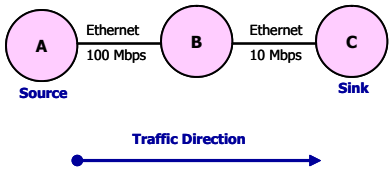

Fig. 2. The D-QoS Prototype

node is triggered to reconfigure its settings for the interruption, the flow payload can be transmitted. At the end of the transmission, another program packet is sent to mark the end of the interrupting flow which in effect removes the interruption of the flow and reconfigures the queueing structure on a D-QoS node accordingly. Even though the program and data transmission is separated in our implementation, the mechanism of combining both program and data parts within an active IP packet can be accomplished as presented in [13,14]. To achieve interoperability between normal and active IP nodes, program part is placed in packet options which is unknown and ignored by normal IP nodes, while data part is carried within packet payloads.

Two queueing mechanisms used in our D-QoS system are Class-Based Queueing (CBQ) [17] and Priority Queue(PQ). They are implemented using the Alternate Queueing (ALTQ) package [18]. In our DiffServ implementation, CBQ is used to represent the bandwidth partitioning and sharing among DiffServ classes. In this prototype, the system provides one EF class for real-time flow and four AF classes for background flows where each class has its own queue with a predefined portion of bandwidth. The structure of the CBQ used in the system is shown in Fig. 3. As EF class was designed for traffic that requires low loss, low delay and low jitter, it has been configured with higher priority comparing to AF classes. We give preference to time sensitive traffic where a limited 75% of the total bandwidth is given to the EF class which means that even if there is no other traffic, the EF class can only occupy 75% of the total available bandwidth.

Each of the four AF classes applies an extended version of Random Early Detection with In and Out (RIO) that provides three drop precedence levels. In contrast to the limitation in the bandwidth consumption of EF class, an AF class may receive excess bandwidth, in the absence of any real-time flows in EF class, up to 100% of the total link bandwidth. Unknown and best effort traffic flows are classified into one of the four AF classes, AF class 3 or AF3 as shown in Fig. 3.

In the interruption mode, this CBQ structure is automatically substituted by priority queue (PQ) allowing 16 priority levels maximum. Our prototype is intended to present a realization of our active node concept rather than to study the system performance. The PQ used in our prototype puts both DiffServ EF and AF classes on the same level, priority level 1, hence differs slightly from the queueing structure shown in Fig. 1. Whenever a D-QoS node

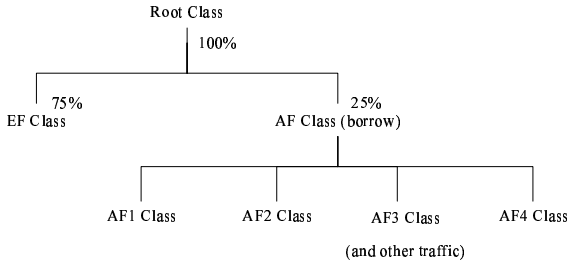

Fig. 3. Class-Based Queuing (CBQ) Structure for DiffServ Implementation

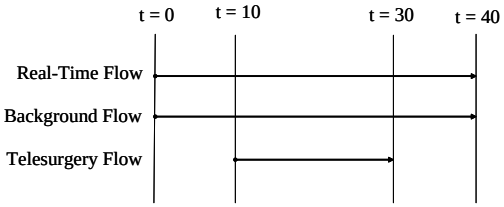

Fig. 4. Traffic Generation Pattern in the Experiments

receives an interruption request, the node reconstructs its queueing structure where a new queue with the requested level is inserted to the PQ allowing a number of privileged flows with different priories to coexist. A queue can be removed once an active packet marking the end of the transmission of a privileged flow is received. Thus, the number of queues and their priorities in this PQ is dynamically adjusted over time according to the active packets received. When there is no more privileged flows in the system, the D-QoS node resumes its normal operation mode with DiffServ. Note that, each queueing reconstruction results in empty queues where the previous queue content has been flushed.

The two Ethernet links in the prototype have been configured with different speeds, 100 Mbps and 10 Mbps, in order to create a bottleneck. The aims of the experiments are to demonstrate how interruption can be realized using a network with D-QoS nodes. We set up an experiment whereby a telesurgery flow which needs high precision of data transmission and also has high bandwidth is authorized to interrupt any existing flows. The interruption request is assumed to be the one at level 16. Similar to the transmission of other time sensitive flows where the transmission rate is preferred over reliability, the flow is transmitted over UDP rather than TCP. In our experiments, three flows representing the telesurgery flow, the real-time flow and the background Internet flow are generated by the generator program, Iperf [19], as uni-directional UDP flows from node A to C. Each of the three flows is a constant rate UDP flow of size 6 Mbps, which consumes 60% of the 10 Mbps link. The flow generation pattern is shown in Fig. 4. The two flows, real-time flow and background Internet flow, are transmitted at the beginning of the experiments and last for 40 seconds. After the first 10 seconds, the telesurgery flow is generated and lasts for 20 seconds.

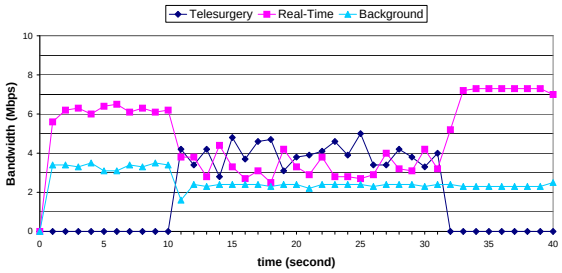

Fig. 5. Experimental Results in Normal Mode with DiffServ

Fig. 5 shows the bandwidth consumed by the three flows in normal operation mode with DiffServ. In this case, both real-time flow and telesurgery flow are classified into EF class and share the available bandwidth of 7.5 Mbps during the middle 20 seconds of the experiment. The rest of the link bandwidth of 2.3 Mbps is given to the background Internet flow placed into AF class 3 (AF3). The packet loss percentages for each flow during the middle 20 seconds are 42% for the telesurgery traffic flow, 44% for the real-time traffic flow and 63% for the background Internet traffic flow. In the last 10 seconds of the experiment, there is a period where the real-time traffic flow obtains about 7.2 Mbps, which is higher than the generated rate and nearly reaches the maximum bandwidth limitation for EF class. The reason is because there are some packets left in the buffer queue during the congestion time and they are forwarded out when more bandwidth becomes available.

The experimental results in the case of an interruption of the telesurgery flow is shown in Fig. 6. The traffic flows are generated with the same pattern as the previous experiment, as shown in Fig. 4. Prior to the transmission of the telesurgery flow at time t=10 seconds, an interruption request is generated and DiffServ based on CBQ is dynamically substituted by PQ for the interruption mode. The results of the first 10 seconds are based on DiffServ and yield the same results as those in the first 10 seconds of Fig. 5. In the middle 20 seconds where the system operates in the interruption mode, the telesurgery flow acquired 6.3 Mbps with no packet losses. Since all DiffServ flows are placed into a single queue with the lowest priority or level 1 in our prototype, the rest of the available bandwidth is shared between the real-time and the background Internet flows with 62.3% and 59.8% packet loss percentages respectively. After the completion of the telesurgery flow transmission, the system then resumes back to its normal mode in the last 10 seconds. As the queue content is discarded, the experiment in this period gives the same results as those at the beginning 10 seconds. Comparing to the result in the normal mode, the interruption mechanism provided in D-QoS allows the telesurgery flow to be delivered with less packet loss.

For the case of multi-level interruption, we assume two interruption requests on an existing flow X (as a background Internet flow). First, an interruption

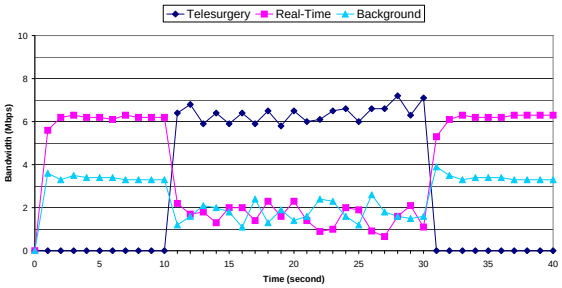

Fig. 6. Experimental Results in Interruption Mode

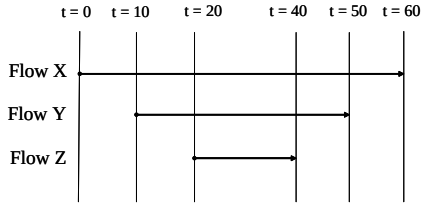

Fig. 7. Traffic Generation Pattern in the Experiments for Multi-Level Interruption

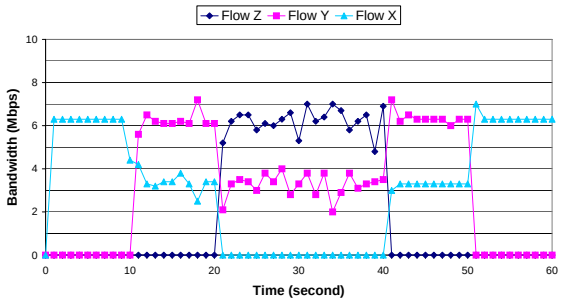

Fig. 8. Experimental Results with Multi-level Interruptions

by privileged flow Y at level 10. Then, another interruption by privileged flow Z at level 16. All flows are of 6 Mbps each. The traffic generation pattern is illustrated in Fig. 7 and the experimental results are shown in Fig. 8. CBQ used in the first 10 seconds is substituted by a PQ with 2 levels when flow Y begins its transmission. PQ with 3 levels is used when the request from flow Z is received. After the completion of flow Z, the queue for flow Z is removed and the system is based on PQ with 2 levels, where DiffServ is resumed at the end. Throughout the experimental period, the results have shown that the flows are transmitted according to their priorities. The highest priority flow gains 6.2 Mbps, while 3.4 Mbps is given to the second priority flow.

4 Performance Analysis

4.1 Simulation Results with Internet Traffic

The performance of a network is decided by those links where congestion occurs, usually called bottleneck links. A necessary condition for our service to be effective is to deploy D-QoS at these links. One can ignore what happens in the other non-congested links of the network. Thus, as a first step, we only consider a single bottleneck topology in our simulations, which follows the D-QoS prototype shown in Fig. 2. The congestion is supposed to appear in one place of the network which is modelled in the topology by the 10 Mbps link, and the rest of the network is abstracted by the two links before and after the bottleneck. In a second step we consider a simulation topology composed of multiple bottleneck links in series.

The simulations are performed on a modified version of ns-2 based on realistic traffic types according to the traffic patterns found in the Internet. For our privileged flow, we use a sample telesurgery flow which is a transmission of a high quality video stream from the patient side to the doctor side. It is represented as a unidirectional VBR stream. On the background, the Internet traffic is divided into two main types, time-sensitive and non time-sensitive applications. Additional video flows with VBR are simulated as time-sensitive background traffic. Non time-sensitive traffic is represented by FTP flows. The performance parameters measured in the simulations for the time-sensitive flows are delay, jitter and packet loss percentage. In case of non time-sensitive flows, only the transmission time is measured.

All VBR flows are generated from video trace files of movies [20–24]. Each trace file records the results obtained from the video encoding algorithm, with ITU-T H.26L standard, and comprises the times at which the frames are transmitted and the frame sizes. These times are then transformed into the inter-frame times, which are equal, of 40 msec (millisecond), since all the streams are generated with the rate of 25 frame/sec. The frame size varies as a result of the video encoding mechanism. Each frame may be split into several UDP packets of 1500 bytes each. As burstiness is not likely to happen in the network due to traffic shaping mechanisms, the packets from a single frame are spaced evenly within 40 msec instead of being generated as a bunch at every 40 msec. Each VBR flow starts with a random starting point within the trace file. As the random starting point is used, the trace files are modified by excluding the beginning and/or ending parts to get rid of the extraordinary frames. When the end of the trace file is reached, the trace is wrapped around and the beginning of the trace is used. To make sure that flows are not synchronized, each flow begins its transmission at a random time within the first 40 msec.

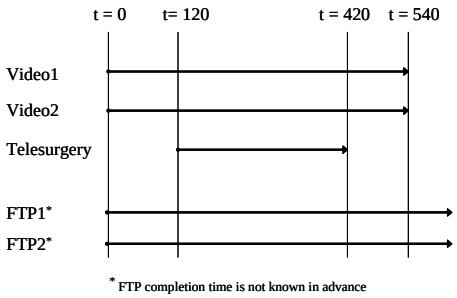

Fig. 9. Traffic Generation Pattern of 5 Flows

The FTP flows are based on TCP Reno implementation in ns-2.

We present, in this section, four simulation scenarios. The first scenario, in Section 4.1.1, aims to compare the performance in normal operation mode where the network uses DiffServ without any D-QoS enabled nodes and in the interruption mode where D-QoS nodes are present. To demonstrate a scenario when the bandwidth available is insufficient, an extreme case where the privileged flow requires higher bandwidth than the total link capacity is reported in Section 4.1.2. The scalability of the system is analyzed in Section 4.1.3 and 4.1.4 based on the number of the interruptions at a particular bottleneck link, and the number of interruption links on a flow path, respectively. For each scenario, a number of simulations are repeated to obtain simulation results with 90% confidence level. As we mainly consider the network within an autonomous system (AS), each link has been configured with 10 msec delay. The sizes of the network queues are set with ns-2 default values of 50 packets.

4.1.1 Performance Comparison with DiffServ

We compare the performance of the flow transmissions in two operation modes, in normal operation mode without the interruption and the one supporting interruption request on D-QoS nodes. The predefined QoS setting follows the CBQ structure previously presented in Fig. 3. The time diagram in Fig. 9 shows the traffic generation pattern used in this scenario. The simulations first start with the transmission of two video flows, video1 and video2, which last for 540 seconds. The telesurgery flow starts 120 seconds after the beginning of the simulation and completes after its 300 seconds of transmission. These flows are generated from a trace file [20], of which the average bit rate is 3.5 Mbps, at different starting points randomly chosen in each simulation run. They are classified into the same class, DiffServ EF class. There are also two FTP flows, each of which is configured for a transfer of 100 MB file. FTP flows start their transmissions at the beginning of the simulation and are placed into DiffServ AF class 3 or AF3.

An example of the bandwidth sharing among the flows, taken from a particular

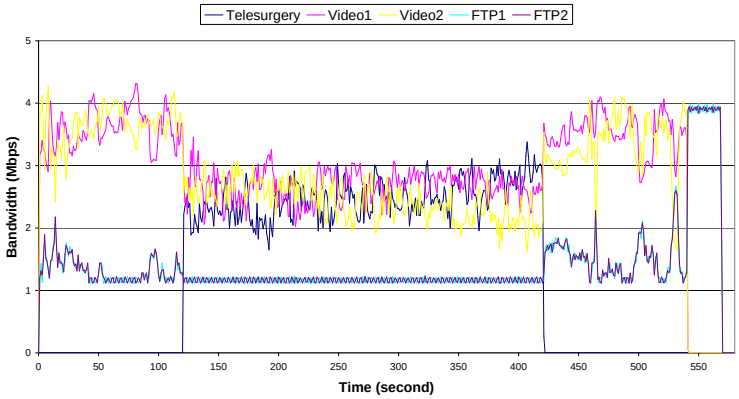

Fig. 10. Bandwidth Sharing of Flows with DiffServ in Normal Mode

Flow	Packet Loss Percentage(%)	Delay(msec)	Jitter(msec)
Telesurgery	27.32±2.35	94.69±0.10	2.11±0.10
Video1	27.16±1.43	94.67±0.10	2.21±0.14
Video2	28.42±1.99	94.70±0.12	2.21±0.14

Table 1
Statistical Results in the Middle 300 Seconds with DiffServ

run in case of DiffServ, is shown in Fig. 10. The bandwidth assigned to EF class of 7.5 Mbps is shared among the three VBR flows and the rest of the link bandwidth is given to the two FTP flows. Table. 1 presents results from 10 simulation runs. The performance parameters are calculated within the 300 seconds transmission time of the telesurgery flow when all flows are active. The performance of the three VBR flows are close to each other as they belong to the same class and, therefore, they receive the same treatment. The two FTP flows, placed in AF3 class, finish their transmissions at nearly the same time. The transmission time used by the FTP1 is 563.24±3.96 seconds, while FTP2 uses 563.26±3.96 seconds.

When D-QoS nodes are present in the network, the telesurgery flow can make an interruption, assumed at level 16, on other existing flows before its transmission. It then receives the highest priority during the interruption period of 300 seconds. The bandwidth sharing result, taken from a single run, is shown in Fig. 11. The rest of the link bandwidth is first given to the two VBR flows previously belonging to EF class, as they have priority level 2. The two FTP flows have priority level 1, the lowest level, and receive nearly no bandwidth during the interruption. The results during the interruption, calculated from 50 runs, are shown in Table. 2. The transmission times for FTP flows are 649.27±1.78 and 649.69±1.69 seconds for FTP1 and FTP2 respectively.

Comparing to the results in the normal mode with DiffServ, the telesurgery flow is transmitted over D-QoS nodes with the best service from the network

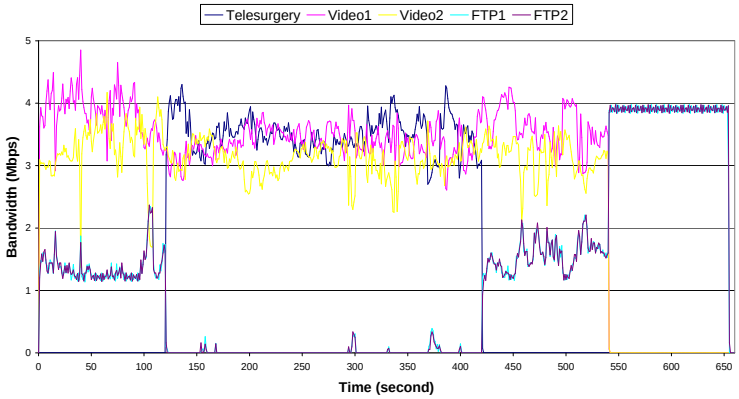

Fig. 11. Bandwidth Sharing of Flows with D-QoS Interruption Mechanism

Flow	Packet Loss Percentage(%)	Delay(msec)	Jitter(msec)
Telesurgery	0.00±0.00	21.86±0.00	0.41±0.00
Video1	6.96±0.66	86.74±3.26	29.82±1.55
Video2	7.24±0.77	86.67±3.27	29.79±1.56

Table 2
Statistical Results in the Middle 300 Seconds with D-QoS Interruption Mechanism

and receives much lower delay and jitter, with no packet loss at all. The performance of the two video flows are also improved with less delay but higher jitter, and less packet loss percentages. D-QoS outperforms DiffServ in this particular case. This is because the static configuration of DiffServ is posing limitation on bandwidth allocation of the EF flows where 7.5 Mbps is given to all the three VBR flows which is insufficient. D-QoS allows dynamic bandwidth allocation where all the three VBR flows share the 10 Mbps link bandwidth, in which the highest priority is given to the telesurgery flow (3.5 Mbps) and the other two flows each of 3.25 Mbps. Thus, dynamic bandwidth allocation scheme in D-QoS offers better services, in terms of delay and loss, to EF flows when the preconfigured bandwidth limitation in DiffServ is insufficient for their requirements. The jitter experienced by these two VBR flows are higher due to the interruption mechanism where the packets are possibly blocked for longer times. Video1 and Video2 have less packet loss percentages since there are two flows in the queue instead of three flows. The two FTP flows need longer transmission times because they are pending during the interruption period. The bandwidth portion of 2.5 Mbps assigned to them in DiffServ is no longer available for their transmissions, but is reallocated to those VBR flows. Most of FTP data are transmitted after the interruption has completed.

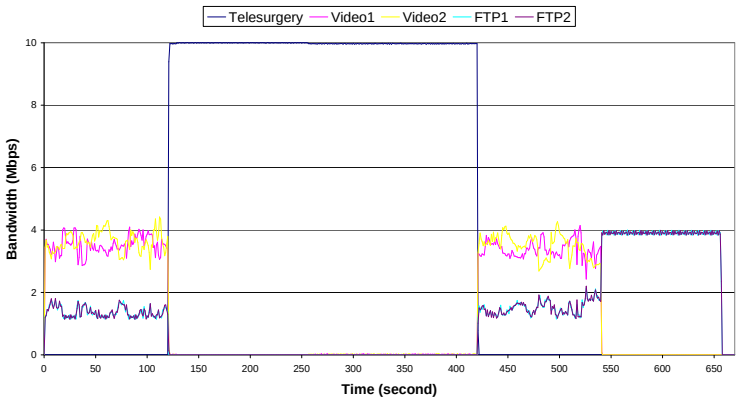

Fig. 12. Bandwidth Sharing in the Extreme Case of the Interruption

Flow	Packet Loss Percentage(%)	Delay(msec)	Jitter(msec)
Telesurgery(14Mbps)	33.38±0.02	79.26±0.01	1.00±0.02
Video1(1Mbps)	99.99±0.00	n/a	n/a
Video2(1Mbps)	99.99±0.00	n/a	n/a

Table 3
Statistical Results During the Interruption Period in the Extreme Case

4.1.2 An Extreme Case

It is possible that the bandwidth requirement of the privileged flow is higher than the link capacity. In this scenario, the telesurgery flow is represented as a 14.3 Mbps VBR flow [21]. Two 3.5 Mbps VBR flows [20] and two FTP of 100 MB files are generated as the background traffic. The simulations follow the time diagram used in the previous scenario, as shown in Fig. 9. The bandwidth sharing result is shown in Fig. 12 and the statistical results during the interruption, from 5 runs, are presented in Table. 3. The results demonstrate that the privileged flow is always given the network resources at D-QoS nodes. It can be seen, from the figure, that no other flows is serviced during the interruption period. However, even the link is dedicated to the telesurgery flow, the flow also experiences losses since its bandwidth requirement is higher than the network capacity. In this scenario, the network queue is always full and, therefore, this results in higher delay and jitter of the telesurgery flow, in comparison to the former case in Section 4.1.1. With nearly 100% packet loss of the two background VBR flows, their delay and jitter cannot be reported. The transmission time of the two FTP flows are 655.63±2.64 and 655.68±2.63 seconds for FTP1 and FTP2 respectively.

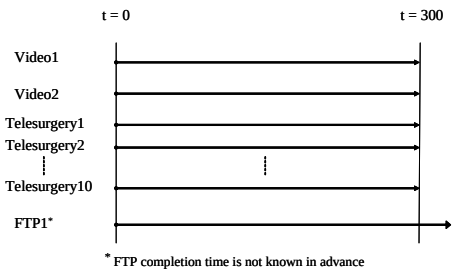

Fig. 13. Traffic Generation with Various Number of Telesurgery Flows

4.1.3 Multiple Number of Interruptions

As Internet traffic is dynamic, there can be an arbitrary number of flows at a bottleneck link. In this scenario we aim to find the number of interruptions of privileged flows at a bottleneck. The background traffic is configured to consume half of the link bandwidth and consists of two VBR flows and one FTP flow. Video1 and Video2 have the average bit rates of 3.4 and 1.9 Mbps respectively [22,23]. The FTP flow is configured for a transmission of a 100 MB file. The telesurgery flow is a VBR flow of size 1.2 Mbps [24]. All flows start their transmissions at the same time at the beginning of the simulation and last for 300 seconds, except the FTP flow in which its completion time is unknown, as illustrated in Fig. 13. We increase the number of telesurgery flows in the system, with an increment of its interruption level, one by one. The first interruption is at level 3 and the interruption level of the 10^{th} telesurgery flow is at level 12. The results of the performance degradation of the background traffic are shown in Fig. 14–17. The two background video flows start to have packet loss when there are 3 telesurgery flows in the system. Packet loss percentages increase linearly until they reach 100%. When the loss percentage reaches 100%, the delay and jitter cannot be reported. The transmission time required by the FTP flow increases and becomes stable after there are 4 telesurgery flows in the system as FTP flow can use the link only after all the video flows have completed.

We can conclude that the maximum number of telesurgery flows at a bottleneck link is limited by the total link bandwidth. All telesurgery flows are transmitted with nearly 0% packet loss and low delay and jitter. However, when the total bandwidth requirement of the privileged flows nearly reaches the bottleneck link bandwidth, the privileged flow with the lowest priority starts to experience performance degradation. In this case, when there are 7 telesurgery flows in the system, even the total average rate is 8.4 Mbps, the first telesurgery flow with the lowest interruption priority (level 3) starts to experience 2.9% packet loss due to the nature of VBR traffic.

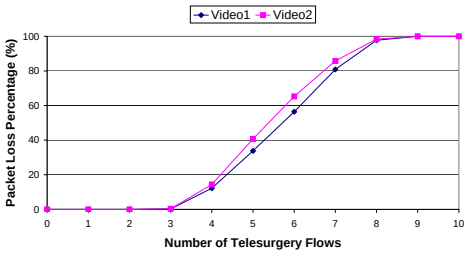

Fig. 14. Packet Loss Percentage of Background Video Flows, with Various Number of Telesurgery Flows

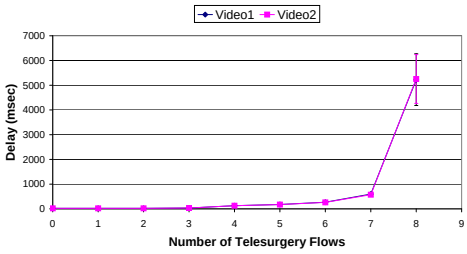

Fig. 15. Delay of Background Video Flows, with Various Number of Telesurgery Flows

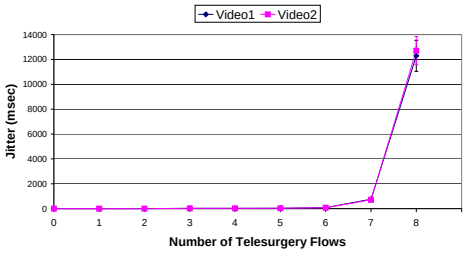

Fig. 16. Jitter of Background Video Flows, with Various Number of Telesurgery Flows

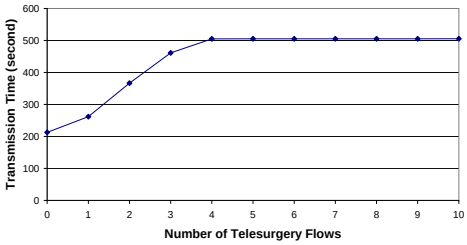

Fig. 17. FTP Transmission Time with Various Number of Telesurgery Flows

4.1.4 Multiple D-QoS Enabled Links

This section presents the impact of the system with multiple interruption links on a flow path. We increased the interruption links on a path by increasing, one by one, the number of D-QoS nodes connecting to the system with the 10 Mbps Ethernet links, each has 10 msec delay. The traffic flows are generated in the same manner as in the previous section, following Fig. 13, but with 3

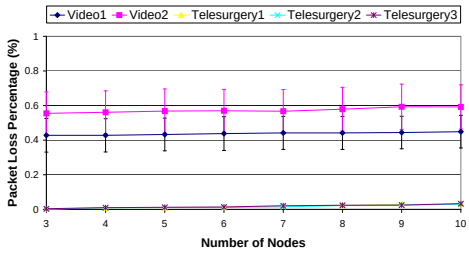

Fig. 18. Packet Loss Percentage of Video Flows, with Various D-QoS Links

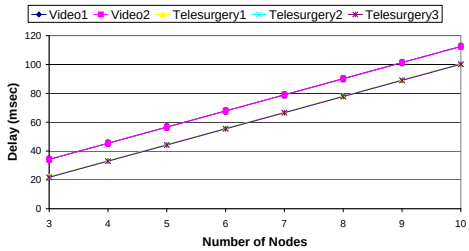

Fig. 19. Delay of Video Flows, with Various D-QoS Links

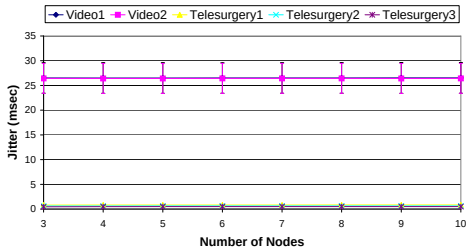

Fig. 20. Jitter of Video Flows, with Various D-QoS Links

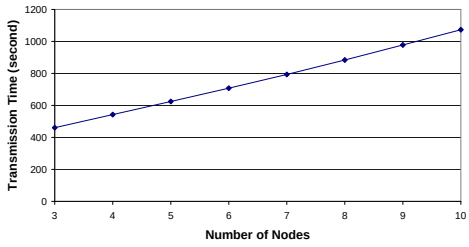

Fig. 21. FTP Transmission Time with Various D-QoS Links

telesurgery flows. The results are shown in Fig. 18–21. From our simulation results, it is observed that by increasing the number of nodes on the flow path does not affect the performance of the privileged flows. The privileged flows are given the best service over all the links with D-QoS nodes. Even the flows experience higher delays, but they are due to the unavoidable link delays. As the link is added linearly with a fixed delay of 10 msec each, the delay of each flow is also linearly increased. Thus, having more D-QoS nodes on the flow path would ensure the transmission quality of the privileged flows.

4.2 Analytical Analysis of Delays

We model the D-QoS mechanism by focusing on a D-QoS router that operates in the Internet backbone. We assume that the D-QoS router degree is large and that each individual flow has a very small throughput compared to the router capacity. On each router there is a buffer, called NQ, for queuing packets coming from the interrupted non-privileged flows such as DiffServ EF, and a set of priority buffers, called IQs, for flows that have priority over the non-privileged flows. The NQ and IQ buffers can be seen as a set of priority queues in parallel ranked in the increasing order of priority. Since the network is large, we assume that there are lot of IQ queues on each D-QoS node and that each D-QoS node only sends a quantum of its load to each of its neighboring routers.

That way, considering an arbitrary router and referring to the Law of Large Numbers we can assume that the global traffic load in its IQ queues and in its NQ queue are Poisson of respective load λ_I and λ_N in packets/s. The input traffic in each prioritized queue may be periodic with jitter but it does not matter since we assume there are so many IQ queues that none of them will contain more than one packet at a time. We further assume that all flows in the network have the same throughput. This assumption is not critical, it is made to ease the exposition of the analysis.

4.2.1 Queueing Analysis

If X and Y are two integer random variables, we denote by $X * Y$ the random variable obtained by summing X identical independent distributed (i.i.d) copies of Y. Or in other words $X * Y = Y^1 + \cdots + Y^X$ where the Y^is are i.i.d like Y. Notice that $X * Y \neq Y * X$. Let $f(z)$ and $g(z)$ be respectively the probability generating functions (p.g.f.) of variables X and Y: $f(z) = \sum_k P(X = k)z^k = E(z^X)$ and $g(z) = \sum_k P(Y = k)z^k = E(z^Y)$, then the p.g.f of $X * Y$ is $E(z^{X*Y}) = f \circ g(z)$

4.2.2 Delay Distribution for Privileged Queues

We consider that the NQ and IQ priorities rank from 0 to 1 with 1 being the lowest priority and 0 the highest priority. Therefore an IQ flow of priority x has to give priority to a volume of traffic of size $\lambda_I x$. We take a slotted model and slot duration is the time unit.

First we analyze the length of the busy period B_x of the traffic of priority less than x, i.e. the waiting time between two consecutive slots accessible for traffic of priority x. We denote by $\beta_x(z)$ the probability generating function (p.g.f.) of B_x, namely $\beta_x = \sum_k P(B_k = k)z^k$. We have the identity between

random variables

$$B_x = A_x * (1 + A_x * (1 + \cdots)) \tag{1}$$

or, similarly

$$B_x = A_x * (1 + B_x) \tag{2}$$

where A_x is the number of packet arrivals of priority less than x during a time slot. Since the p.g.f of A_x is $\exp((z-1)x\lambda_I)$, we have the p.g.f identity.

$$\beta_x(z) = \exp((z\beta_x(z) - 1)x\lambda_I) \tag{3}$$

Using the well known Cayley Tree function $T(z) = \sum_k \frac{k^{k-1}}{k!} z^k$ which is the solution of the functional equation $T(z)\exp(-T(z)) = z$ [25] we get

$$\beta_x(z) = T(x\lambda_I e^{-x\lambda_I} z)\frac{1}{x\lambda_I z} \tag{4}$$

We have $P(B_x = k) = (\frac{(k+1)^k}{(k+1)!}\left(x\lambda_I e^{-x\lambda_I}\right)^k$. Since the tree function has a singularity on $z = e^{-1}$ which is of the square root type, we have $P(B_x = k) \sim \frac{1}{k\sqrt{2\pi k}}\left(x\lambda_I e^{-x\lambda_I+1}\right)^k$. Notice that $x\lambda_I e^{-x\lambda_I+1} < 1$ as long as $0 < x\lambda_I < 1$.

Second, from the busy period expression we derive the p.g.f of the delay distribution W_x of a random packet of the flow of priority x. Taking $w_x(z) = \sum_k P(W_x = k)z^k$, it comes that W_x is the remaining of a busy period B_x taken at a random time. In other word:

$$\sum_k P(W_x = k)z^k = \frac{\sum_k P(B_x = k)(1 + z + \cdots + z^{k-1})}{\sum_\ell P(B_x = \ell)\ell} \tag{5}$$

$$= \frac{1 - \beta_x(z)}{(1-z)\beta'_x(1)} \tag{6}$$

and it has the same singularity as $\beta_x(z)$, indeed one finds $P(W_x = k) \sim \frac{1}{\sqrt{2\pi k}}\left(x\lambda_I e^{x\lambda_I+1}\right)^k$. The delay distribution of packets belonging to IQ flows has a geometric tail (the factor $\sqrt{k}$ has negligible variations compared to exponential).

4.2.3 Delay Analysis of the Non-Privileged Queue

The non-privileged queue, NQ, is equivalent to a single server queue with vacation. A vacation corresponds to the busy period of the whole IQ queues,

namely B_1 plus one slot.

First, let Q be the queue length after the end of a random busy period. We have the random variable identity obtained from the evolution over a vacation period:

$$Q = Q - 1|^+ + (B_1 + 1) * A_E \tag{7}$$

where $x|^+ = x$ if $x > 0$ and zero otherwise, and A_E is a per slot packet arrival in the NQ queue. The p.g.f of A is $a(z) = \exp((z-1)\lambda_N)$.

Translating this identity in p.g.f., with $q(z) = \sum_k P(Q = k)z^k$ it comes:

$$q(z) = \left(\frac{1}{z}(q(z) - q(0)) + q(0)\right) a(z)\beta_1(a(z)) \tag{8}$$

and therefore

$$q(z) = q(0)a(z)\beta_1(a(z))\frac{z-1}{z - a(z)\beta_1(a(z))} \tag{9}$$

and $q(0) = 1 - (\beta_1'(1) + 1)\lambda_N$ in order to keep $q(1) = 1$ (unitarity). It turns out that $P(Q > k) \sim \rho_1^{-k}$ where ρ_1 is the first root greater than 1 of $z - a(z)\beta_1(a(z))$.

Second we look at the delay that a packet experiences when it arrives during a vacation period of length ℓ. The delay is $Q * (B_1 + 1) + \Omega_\ell$ where Ω_ℓ is the remaining delay, *i.e.* the sum of the remaining time before the end of the current vacation period and the additional vacation cycles the packet has to wait due to the packets arrivals before its own during that vacation cycle. Let Ω_ℓ^j be the remaining delay when packet arrives on the jth slot of a vacation period of length ℓ. We have $\Omega_\ell^j = \ell - j + (j-1) * A_E * (B_1 + 1) + C_E * (B_1 + 1)$, where C_E is the number of arrivals on slot j which arrived before the packet. The p.g.f of C_E is $c(z\beta_1(z)) = e^{(z-1)\frac{\lambda_N}{2}}$. Therefore $\omega_\ell^j(z) = E(z^{\Omega_\ell^j}) = c(z\beta_1(z))z^{\ell-j}(a(z\beta_1(z)))^{j-1}$. Let $\omega_\ell(z) = \sum_k P(\Omega_\ell = k)z^k$. We have

$$\omega_\ell(z) = \frac{1}{\ell}\sum_{j=1}^{\ell} \omega_\ell^j(z) \tag{10}$$

$$= \frac{1}{\ell} c(z\beta_1(z)) \frac{a(z\beta_1(z))^\ell - z^\ell}{a(z\beta_1(z)) - z} \tag{11}$$

The p.g.f. of $Q * (B_1 + 1)$ is equal to $q(z\beta_1(z))$. Let $\omega(z)$ be the unconditional p.g.f of the remaining delay. we have:

$$\omega(z) = \frac{1}{1+\beta'(1)} \sum_{\ell} \ell P(B_1 + 1 = \ell)\omega_\ell(z) \tag{12}$$

$$= c(z\beta_1(z)) \frac{(a(z\beta_1(z))\beta_1(a(z\beta_1(z))) - z\beta_1(z))}{(1+\beta'(1))(a(z\beta_1(z)) - z)} \tag{13}$$

Therefore the full delay p.g.f is equal to $q(z\beta_1(z))\omega(z)$. The main singularity comes from the singularities of $q(z\beta_1(z))$, *i.e.* the root of function $z\beta_1(z) - a(z\beta_1(z))\beta_1(a(z\beta_1(z)))$. In fact we have $\rho_1 = \rho_2\beta_1(\rho_2)$: ρ_2 maps to ρ_1 with the function $z \to a(z\beta_1(z))$. The main singularity being a simple pole we have therefore $P(W = k) \sim \rho_2^{-k}$, which is a geometric tail of rate ρ_2^{-1}.

4.2.4 Delay Analysis in Full Network and Equivalent Bandwidth

According to our basic hypotheses, the routers are independent. In this case the p.g.f of the overall delay in the network for a given flow is equal to the product of the p.g.f of the delays that are experienced in the routers on the path of the flow. Since the delay probability distribution have geometric tail, the overall delay has also a geometric tail (which is basically the tail of the distribution in the router that provides the dominant tail). This refinement allows us to define an equivalent bandwidth for an IQ flow when we have to conform to probabilistic delay requirement of the form $P(\text{delay} > T)$ should be smaller than a given ϵ. For detailed definition and discussion about equivalent bandwidth, the reader should refer to the excellent survey [26]. For the IQ traffic one must take care of the fact that the priority x of a given flow may not be the same on the different routers crossed by the flow and therefore for the same path and same delay requirement, two flows may have two different equivalent bandwidths.

4.3 Security and Scalability

In this paper, we consider the use of D-QoS within an AS, thus the security issue is not prominent and is outside the scope. However, the works in [27–29] has suggested the implementation of security in active networks where the sending network element or user identification and resource access authorization are addressed for secure execution of active programs. The programs should be constructed based on the languages that provide safe execution such as Java and a special language particularly designed for active networks in [30], and run on a secure environment with limited resource (CPU time and memory). User identification is verified through cryptographic authentication mechanism based on principal ID and digital signature to allow only trusted users to perform the executions on network nodes. To limit the resource consumption, node resource access is allowed only through the interfaces. Another

secured active network has been proposed by [31] based on another idea where a web of trust elements is maintained using public/private key authentication.

In our D-QoS model, accesses to resources within each node is limited to interruption handling operations through primitive invocation. Nevertheless, a privileged user authentication is important and should be addressed. For large scale deployment, D-QoS boundaries could be set up whereby user authentication can be applied to incoming active packets at the boundary nodes. Once entered into the secured domain, active packets from privileged users can be freely admitted to reconfigure the network nodes within the boundary. This is similar to DiffServ in which complex functions are performed at network boundaries and does not cause overhead on the interior nodes.

Although D-QoS provides QoS on a per-flow basis, similar to IntServ, the overhead for node processing at each network node is minimal where the flow state information does not have to be maintained. With an increased number of D-QoS links on a given path, it would strengthen the assurance of the transmission quality for the privileged flows. The number of D-QoS links on the flow path has been shown to have no effect on the throughput, except for some extreme cases where link capacity is insufficient.

Since the D-QoS node is an active IP node, the D-QoS scheme could be interoperated with normal IP nodes in the network. The network administrators may deploy D-QoS nodes, at the beginning, only in a part of an edge network where the link bandwidth is tight. The interruption mechanism provided by D-QoS would allow the privileged flow to be transmitted over the tight bandwidth link with the highest priority. Then the administrator may subsequently expand the number of D-QoS nodes to cover other links within the network.

The performance of the active nodes is reported in [27] with consideration on per capsule forwarding, code distribution and periodic node management tasks. The capsule processing within active nodes in [27] is at most a factor of two of the basic processing time.

For scalability, though processing requirements for each active node is much lass than those of IntServ, it is suggested that D-QoS should be partially deployed. Network nodes without tight bandwidth problem that are highly loaded, such as core routers, should be normal IP nodes and thus are not slowed down by capsule processing. With the advances of the underlying technology, the bottleneck routers are considered having limitation on bandwidth rather than the processing capacity. Hence, implementing D-QoS over these nodes would not affect the throughput of the routers.

5 Related Works

Dynamic resource allocation presented in [6] differs from D-QoS in which the resource allocation is free from any existing preconfiguration. The QoS setting in D-QoS is automatically adjusted to offer the interruption according to the requests from privileged flows. In addition, our work also differ from theirs in term of service granularity while their work offers QoS based on an aggregation of flows basis, our proposed D-QoS provides services based on a per flow basis where interruption is generated for a particular flow.

In [9], a dynamic node configuration architecture, namely DACA architecture, was proposed using active network technology for the control of DiffServ. Similar to DACA, D-QoS model employ active network with the capsule approach, to provide a QoS reconfigurable network. However, the idea of QoS reconfiguration differs in its details where in DACA architecture, it is required that the network administrator adjusts the configuration, through the changes of policies, according to the modification in SLA or in response to the traffic monitoring process. In D-QoS, the QoS reconfiguration can be performed automatically once the privileged application starts its transmission without involving a network administrator. D-QoS also allows the network to quickly react to the changes in the traffic requirements or the network condition when compared to the reconfiguration in [9] where the SLAs may be modified in a daily, weekly or monthly basis.

6 Conclusion

The proposed D-QoS system allows QoS settings to be reconfigured dynamically according to user requests, based on the active network concept. An interruption request by an authorized user can automatically trigger QoS reconfiguration on D-QoS nodes. The privileged flow is then transmitted over the network with the highest priority at the expense or possibly blockage of other existing flows. Through both experimentation and simulation, the system has been shown to work satisfactorily where the QoS settings are correctly reconfigured according to the interruption requests and the privileged flows are transmitted with high priorities. In an extreme case in which the bandwidth requirement of the privileged flow is more than the link capacity, all the network resources are given to this flow and the link is dedicated to the flow while other flows of lower priorities are blocked. In addition to providing high quality of service to privileged flows, the system can also improve the bandwidth utilization which may result in better transmission performance of other flows.

With the D-QoS prototype and its performance evaluation, we have shown that the active networking concept can be used as a tool to achieve dynamic QoS network where network nodes can automatically adjust their settings according to the active packets received. Even the model provides QoS based on a simple priority queue mechanism, more general and complex QoS models may also be accomplished through the use of active networks in the future.

7 Acknowledgements

The first author wishes to thank the Royal Thai Government for providing a scholarship for her study. This work was partially supported by the French regional cooperation programme and the 2004 Next Generation Internet STIC-ASIE project.

References

[1] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, G. J. Minden, 'A Survey of Active Network Research', IEEE Communications Magazine, Vol. 35, No. 1, January 1997, pp. 80-86

[2] R. Braden, D. Clark, S. Shenker, 'Integrated Services in the Internet Architecture: an Overview', RFC 1633, June 1994.

[3] A. Detti, M. Listanti, S. Salsano, L. Veltri, 'Supporting RSVP in a Differentiated Service Domain: an Architectural Framework and a Scalability Analysis', ICC'99, June 1999.

[4] Y. Bernet, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, E. Felstaine, RFC 2998, 'A Framework for Integrated Services Operation over Diffserv Networks', November 2000.

[5] S. Salsano, E. Sangregorio, M. Listanti, 'COPS DRA: a protocol for dynamic Diffserv Resource Allocation', Joint Planet-IP NEBULA workshop, Courmayeur, Italy, January 2002.

[6] A. Ramanathan, M. Parashar, 'Active Resource Management for The Differentiated Services Environment', in Proc. of the Third Annual International Workshop on Active Middleware Services, IEEE Computer Society 2001, 2001, pp. 78-86.

[7] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, 'An Architecture for Differentiated Services', RFC 2475, December 1998.

[8] Y. Bai, M. R. Ito, 'Active Network-based Mechanisms and Node Architecture to Enhance Quality of Service for Video Transport over IP Networks', 5th Workshop on Media and Streaming Processors (MSP5), December 2003.

[9] N. Achir, N. Agoulmine, M. Fonseca, Y. Ghamri-Doudane, G. Pujolle,'Active Technology as an Efficient Approach to Control DiffServ Networks: the DACA Architecture', IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS) 2002, October 2002.

[10] T. Tansupasiri, K. Kanchanasut, 'Dynamic Quality of Service on IP Networks', in Proc. of ICOIN 2003, LNCS 2662, Springer-Verlag, 2003, pp. 573-582.

[11] 'ABone Home page', `http://www.isi.edu/abone/`.

[12] 'Network Simulator – ns-2', `http://www.isi.edu/nsnam/ns/`.

[13] D. J. Wetherall, D. L. Tennenhouse, 'The ACTIVE IP Option', in Proc. of the Seventh ACM SIGOPS European Workshop, September, 1996.

[14] D. M. Murphy, 'Building an Active Node on the Internet', Master's thesis, MIT, 1997.

[15] B. Davie, A. Charny, J. C. R. Bennett, K. Benson, J. Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, D. Stiliadis, 'An Expedited Forwarding PHB (Per-Hop Behavior)', RFC 3246, March 2002.

[16] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, 'Assured Forwarding PHB Group', RFC 2597, June 1999.

[17] S. Floyd, V. Jacobson, 'Link-Sharing and Resource Management Models for Packet Network', IEEE/ACM Transactions on Networking, Vol. 3 No. 4, August 1995.

[18] K. Cho, 'A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers', Proceedings of USENIX 1998 Annual Technical Conference, 1998.

[19] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, 'Iperf', May 2002, `http://dast.nlanr.net/Projects/Iperf/`

[20] Acticom Mobile Networks, 'Starship Troopers: Quantization parameter for all frames: 1', 2002, `http://www.acticom.info/1469.html`

[21] Acticom Mobile Networks, 'Bridge Close (CIF): Quantization parameter for all frames: 1', 2002, `http://www.acticom.info/1631.html`

[22] Acticom Mobile Networks, 'Bridge Close (QCIF): Quantization parameter for all frames: 1', 2002, `http://www.acticom.info/1630.html`

[23] Acticom Mobile Networks, 'High Way (QCIF): Quantization parameter for all frames: 10', 2002, `http://www.acticom.info/1572.html`

[24] Acticom Mobile Networks, 'Starship Troopers: Quantization parameter for all frames: 10', 2002, `http://www.acticom.info/1467.html`

[25] J. Fill, P. Flajolet, N. Kapur, 'Singularity Analysis, Hadamard Products, and Tree Recurrences', Journal of Computational and Applied Mathematics, Vol. 174, February 2005, pp. 271-313.

[26] L. Georgiadis,P. Georgatsos, K. Floros, S. Sartzetakis, 'Lexicographically optimal balanced networks', IEEE/ACM Transactions on Networking, Vol. 10, Issue 6, December 2002, pp. 818-829.

[27] D. Wetherall, 'Active Network Vision and Reality: Lessons from a Capsule-based System', 17th ACM Symposium on Operating Systems Principles, December 1999, pp. 64-79.

[28] S. Murphy, E. Lewis, R. Puga, R. Watson, R. Yee, 'Strong security for active networks', In The Fourth IEEE Conference on Open Architectures and Network Programming, April 2001.

[29] T. Faber, B. Braden, B. Lindell, S. Berson, K. Bhasker, 'Active Network Secutiry for the ABone', November 2001.

[30] M. Hicks, P. Kakkar, J. T. Moore, C. A. Gunter, S. Nettles, 'PLAN: A Packet Language for Active Networks', in Proc. of the International Conference on Functional Programming (ICFP) 1998, 1998.

[31] D. S. Alexander, W. A. Arbaugh, A. D. Keromytis, J. M. Smith, 'A Secure Active Network Environment Architecture', IEEE Network Magazine, Special Issue on Active and Controllable Networks, 1998.