MODELING AND PRELIMINARY SIMULATION STUDIES FOR PACKET-BASED PRECEDENCE AND PREEMPTION FOR FCS COMMUNICATIONS

Robert G. Cole and Philip F. Chimento
JHU Applied Physics Laboratory
{robert.cole,philip.chimento}@jhuapl.edu *

27 September 2006

ABSTRACT

In military networks, there had long been a requirement that some traffic be given precedence over other traffic. In the past, this has applied primarily to connection-oriented communication such as voice calls. With the advent of all-IP networks, applying the requirement to connectionless data, while maintaining QoS characteristics necessary for the correct operation of applications, presents challenges. In this paper, we describe our preliminary study of an approach to handling high-precedence IP traffic node-by-node.

1. INTRODUCTION

The U.S. Army's Future Combat System (FCS) is reliant upon the development of a reliable, resilient communications capability under harsh, battlefield environments. During periods of crisis, the communications infrastructure must be capable of providing preferential delivery of information based upon the Future Force Warrior's indication of the importance of the information. In current circuit-switched networks, this is indicated by the message precedence level. The Joint Chiefs of Staff (JCS) have developed instructions for the Global Information Grid (GIG), and hence FCS communications, for support of Precedence and Preemption (P&P) capabilities in (CJCSI, 2004). These instructions extend the traditional telephony Multi-Level P&P (MLPP) services to all C2 messages and applications. In order to support these P&P requirements, the GIG all-Internet Protocol (IP) packet-based transport network must develop new packet handling and forwarding algorithms to simultaneously support application Quality of Service (QoS) and content Precedence Level (PL). Work exists in the literature on the design of forwarding algorithms, commonly referred to as Per Hop Behaviors (PHB) to meet the QoS requirements of applications, e.g., (Keshav, 1997). However, to date, no work exists to design and investigate PHB algorithms which simultaneously deliver QoS to applications and P&P transport to information.

Experience in providing P&P capabilities in communications services fall into two camps, i.e., traditional telephony services, e.g., the Defense Switched Network (DSN), and message handling services, e.g., the Automatic Digital Network (AUTODIN). Naively mapping these onto an all-IP, packet-based transport network like the GIG is problematic. The DSN handled high PL traffic through signaling to indicate the precedence level and resource reservation for assured call set-up. AUTODIN provided preferential queuing and scheduling to high precedence level messages. The difference in IP networks is that under *diffserv* QoS is handled on a packet-by-packet basis, not connection-by-connection because *diffserv* is intended to provide QoS for aggregates. It is possible to have a "connection-oriented" approach to providing QoS, using *intserv* or *MPLS*, but these also have costs in the amount of state that intermediate nodes must keep. The use of these protocols can also be problematic in the highly unstable tactical network.

As indicated in (Liebowitz, 2005), the packet handling must provide preferential transport to high precedence traffic under all networking conditions, specifically conditions of resource scarcity, e.g., network overload conditions, while simultaneously satisfying packet scheduling required to meet application QoS needs. One approach to this requirements duality is to enhance Active Queue Management (AQM) techniques to handle P&P requirements and rely upon standard, well studied QoS PHB, e.g., Weighted Round Robin, Class-Based Fair Queuing, etc., for handling QoS requirements. In this way, when operating under engineered loads the well known scheduling algorithms support high quality QoS for applications. And under network congestion situations, the enhanced AQM layer provides the necessary P&P providing preferential packet handling to high PL infor-

*R. Cole is with the Research and Development Technology Center, and P.Chimento is with the Applied Information Sciences Department, JHU/Applied Physics Laboratory, 12000 Johns Hopkins Road., Laurel, Maryland. 22104

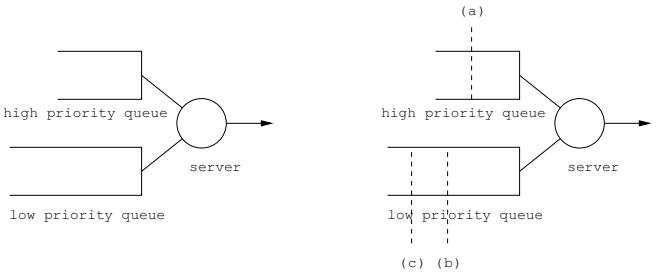

Figure 1: Priority Queue Structure

mation.

The tricky part to developing an enhanced AQM scheme for P&P handling is to prevent the possibility of *precedence inversion* from happening. Precedence inversion occurs when low precedence, delay and jitter sensitive application traffic overloads the communications resource causing high precedence, non-delay sensitive traffic to be discarded. In order to avoid this situation, the enhanced AQM capability must act across the entire set of interface queues and not act within individual class queues. We propose a simple and relatively straightforward scheme for coordinating packet queue admissions across all queues. Our scheme allows low order queues (within the context of QoS handling) to plead up to the next higher order queue for help in alleviating queue congestion under periods of communication link overload. We refer to our scheme as the Cross Queue AQM (CQ-AQM) Scheme. This is illustrated in Figure 1 in the context of a two queue scheme. In the figure, the thresholds indicated by (a) and (b) are local thresholds which trigger discarding of lower PL traffic. The threshold indicated by (c) is a non-local threshold which causes the higher priority queue to discard low PL traffic regardless of the current state of the higher priority queue. Our scheme can be extended to higher numbers of queues.

2. ANALYTIC MODELS

One of the most frequent suggestions for how one could implement precedence levels is to place all the high PL traffic on the highest priority queue and to service that traffic before everything else. In this section, we review some basic properties of priority queue systems and give some quantitative evidence why that naive approach might not produce desired results.

The elementary theory relating to priority queues is presented well in (Kleinrock, 1976). We focus on non-pre-emptive priority systems where packets are served in first-come-first-serve (FCFS) order within each priority. The analysis given in (Kleinrock, 1976) looks at the queue from the point of view of a new packet arriving for service.

Following (Kleinrock, 1976), we assume that there are Q priorities that are numbered from 1 (lowest) to Q (highest). The arrival rate of priority q customers is λ_q. The first moment of the packet length (i.e. service time) for a priority q stream is denoted by $\bar{x}_q$ and the second moment (variance) is denoted by $\bar{x^2_q}$. The utilization contributed by each priority is given by $\rho_q = \lambda_q \times \bar{x}_q$. The analysis assumes that the packet arrival rates are Poisson, but that the packet length distributions can be arbitrary (but independent). We also note that in this theoretical analysis the queues are assumed to be infinite, so there are no expressions for packet loss.

The mean waiting time of a packet arriving at a priority system can be divided into two parts: The time to finish the packet already in service when the new packet arrives and the time to service all the packets already in the same priority or higher priority queues including all the higher priority packets that arrive during this time. In (Kleinrock, 1976) the first part of the average waiting time is given by:

$$W_0 = \sum_{i=1}^{Q} \frac{\lambda_i \bar{x_i^2}}{2} \tag{1}$$

and the waiting time for each priority is given as:

$$W_p = \frac{W_0}{\left(1 - \sum_q^Q \rho_i\right)\left(1 - \sum_{q+1}^Q \rho_i\right)} \tag{2}$$

There are two things to notice about Equations (1) and (2). The first is that they depend on the second moment–that is, the variance–of the packet length, and second, they depend on the queue utilization due to streams at priorities q or *above*.

In order to get a practical idea of what these equations mean we used Equations (1) and (2) to compute the mean waiting time of packets at each of 4 priorities. The priorities are numbered 1 to 4 with 4 being the highest priority. Arrival rates were computes to achieve a given utilization, while the packet sizes were obtained partly from literature and partly from Internet traces. We describe the traces used in more detail in Section 3.3. The results are show in Figures 2 - 5. In this model, the lowest two priorities represented data flows into the queues, priority 3 (second highest) represented video flows and the highest priority represented real-time traffic. Each priority accounted for 25% of the utilization. The only variation between the four figures is the variance of the highest priority flow. We varied this from 0 variance (i.e. constant packet lengths) to a very large variance. The variance was increase artificially, not directly from measured values

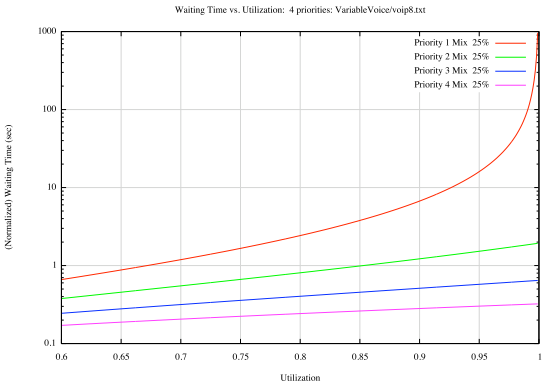

Figure 2: Zero Variance High Priority

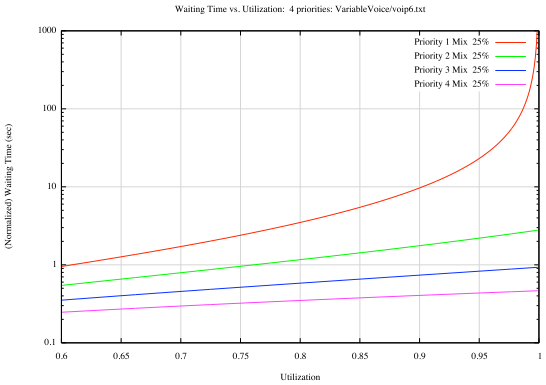

Figure 3: Low Variance High Priority

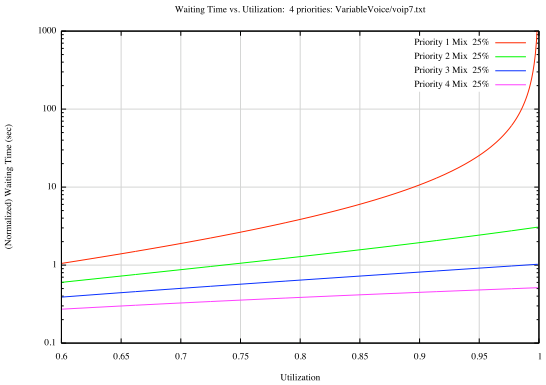

Figure 4: Moderate Variance High Priority

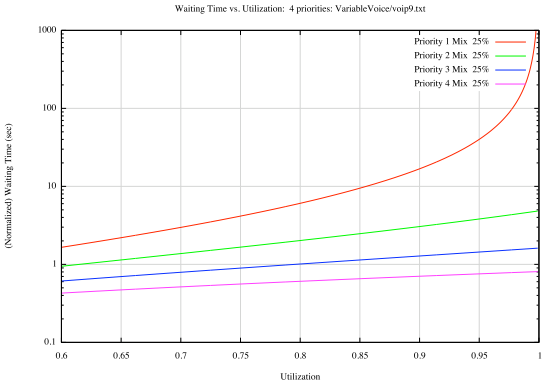

Figure 5: High Variance High Priority

for a flow with that average packet size. See Table 1 for the values used. We can see that based solely on the

		Zero	Low	Mod	High
Pri 1	Mean	1037.64	1037.64	1037.64	1037.64
	Var	426184.53	426184.53	426184.53	426184.53
Pri 2	Mean	755.73	755.73	755.73	755.73
	Var	454120.57	454120.57	454120.57	454120.57
Pri 3	Mean	721.33	721.33	721.33	721.33
	Var	233302.44	233302.44	233302.44	233302.44
Pri 4	Mean	240.12	240.12	240.12	240.12
	Var	0.0	141641.85	189241.59	481879.81

Table 1: Parameters for theoretical analysis

variance of the highest priority class, there is significant increase in the waiting time for *all the other priorities as well as the highest priority class itself*. Even though the average packet size is rather small for the highest priority flows, as the variance increases, the probability of large packets arriving, as well as small packets, not only increases the jitter of the highest priority flow, but also the average delay. This implies that in addition to degrading lower priority service, putting flows with large packet variances all at the highest priority also degrades the service at that priority. This runs counter to the suggestion for handling high precedence traffic at the beginning of this section.

The conclusion that we draw from this is that we needed to search for some other mechanism to ensure that high precedence traffic is preferred in network nodes.

3. SIMULATION STUDIES

We developed a simulation model in order to assess the performance of our CQ-AQM scheme for handling precedence treatment. Because this represents an initial study of new per hop behaviors, we were interested in assessing its performance under a broad range of traffic models, queueing arrangements, scheduling algorithms and drop policies. We felt the best way to accomplish this is to begin with a small simulation model of a single per hop behavior. In this section we describe our methodology, simulation model, arrival and service processes and scheduling and drop policies investigated in our initial performance studies.

3.1. Methodology

The goal of this work is to investigate the ability of the CQ-AQM scheme to protect high precedence traffic from being discarded while low precedence traffic

is served. We have chosen to focus on the impact of design decisions on loss probabilities for various traffic types as the performance metric of study. An extensive literature exists which addresses the relevant performance of various scheduling and drop policies with respect to the QoS supported. Hence, we concentrate on the impact of the new CQ-AQM scheme, when overlaid on a well-known scheduler. To simplify our analysis, we assume only two levels of QoS, i.e., $q = 1$ indicating high priority class, and $q = 0$ indicating low priority treatment. "Priority treatment" can mean strict priority, or preferential scheduling based upon a Weighted Round Robin or Deficit Round Robin scheme. We assume only two levels of precedence, i.e., $p = 1$ indicating high importance, and $p = 0$ indicating low importance. Associated with each QoS class is a queue, which is serviced according to the specific scheduler under consideration. We then compare the loss performance of the various traffic types, i.e., $(p, q) = (0, 0)$, $(0, 1)$, $(1, 0)$, and $(1, 1)$, for the case of no AQM versus the case of having the CQ-AQM scheme. We also track the system delay performance for the various traffic types. For simplicity, we compute our metrics of interest as seen by the arriving packets, hence we present packet averages for loss and delay.

3.2. Simulation Model

For this work, we wrote a small custom simulation program in C++. The structure is that of a simple discrete event simulator with event heap and objects implementing the distributions that drive a given simulation run. We have the capability of instantiating as many objects as necessary to achieve a given utilization level at the queue. The fact that the simulator is a custom program allows us to implement non-standard queue management mechanisms and to have exact control over what information is collected in the course of the simulation. It also allowed us to incorporate objects implementing the empirical distributions very easily. The program is a work-in-progress and as we continue with this analysis, it will be expanded to include additional AQM and scheduling features.

3.3. Arrival and Service Processes

Two different sets of processes were used to drive the simulation experiments: The first set was composed of two different processes to simulate voice over IP (VoIP) and data. The second set was derived from traces collected by the National Laboratory for Applied Network Research (NLANR).

The first set of arrival and service processes consists of a CBR-based VoIP traffic generation model and a highly variable (bursty) process driven by a Markov Chain (Trivedi 1982). The CBR model uses a constant packet size of 68 bytes and is an on-off process. The on-times are 200 milliseconds, and the off-times are 133 milliseconds, approximating talk-spurts and silence periods. The packet generation rate when the model is in an on-state is one packet every 20 ms, which is a typical packetization interval for codecs used for digitized voice. The data stream is modeled by an exponentially distributed packet size with a mean of 100 bytes, and an arrival process that alternates between two states, according to a Markov Chain. The states are a bursty state, where the interarrival time is 460μseconds and a lower-intensity state where the interarrival time is 46 milliseconds. In the high-intensity state, the probability of transitioning to the low-intensity state is 0.011 and the probability of going from the low-intensity state to the high-intensity one is 0.091.

Loc	Day	EST	Samples	Ave IAT	Ave Pkt
AMP	9-10-05	20:54	119743	0.000744	268.56
MEM	3-28-06	10:17	112011	8.11×10^{-5}	417.45
PSC	3-28-06	16:20	888365	1.01×10^{-4}	574.42
MEM	3-28-06	16:26	107137	8.49×10^{-4}	318.98
PSC	3-28-06	13:43	927257	9.69×10^{-5}	549.95
PSC	3-28-06	10:07	1033803	8.67×10^{-5}	600.97
MEM	3-28-06	13:51	117910	7.71×10^{-4}	438.62
PSC	1-14-06	22:54	801054	0.000112	515.98
PSC	1-15-06	07:44	1156601	7.77×10^{-5}	383.57
PSC	1-15-06	13:58	1083000	8.29×10^{-5}	629.75

Table 2: Parameters for empirical distributions

The National Laboratory for Applied Network Research (NLANR) was funded by the NSF and supplied researchers with (among other things) traces of Internet data taken at locations across the US. Many of the traces that were regularly collected were 90 second samples from links between routers connecting GIGAPOP locations. The samples that we used were taken from traces collected from Miami, Florida (AMP) which is an OC-12 link, Memphis, Tennessee (MEM), and OC-3 link and Pittsburgh, Pennsylvania (PSC), and OC-48 link. Table 2 gives a summary of each of the traces. The collection times are given as EST, even though the Memphis site is one hour behind.

The empirical distributions were derived from analyzing the traces and constructing histograms of the inter-arrival times and the packet lengths. For the inter-arrival time distributions, we used a histogram bin size of 2.5 μseconds. For the packet length distributions, we used a bin size of 1 byte. So, in constructing the histograms, some information was lost in the inter-arrival times, but no information was lost for the packet lengths. We did not extract specific application

flows from these traces, but rather used the traces as they were; i.e. as aggregate traffic.

Empirically-derived distributions show some surprising characteristics that are not ordinarily captured by using theoretical traffic models. Both the packet length and the inter-arrival time distributions vary from site to site and from time to time. Most of the packet length distributions are multi-modal, with at least 2 modes, and sometimes as many as 4 or 5. The interarrival time distributions have different shapes, depending on what applications were current on the link when the trace was taken. In addition, the empirical distributions sometimes show very high variance and longer tails than might be seen with standard distributions.

3.4. Scheduling and Drop Policies

Our approach is to assume a given PHB model designed to meet application QoS requirements and then modify the active queue management strategies to incorporate military-unique P&P packet handling without modifying the underlying QoS scheduling mechanisms.

One example of this approach is illustrated in Figure 1. Here we have assumed that the QoS requirements have determined that a two queue, strict priority scheduler is used. Jitter sensitive traffic such as voice is placed in the high priority queue (tagged $q = 1$) and is given strict priority service. Non jitter sensitive traffic, e.g., data, is placed in the low priority queue (tagged $q = 0$). In this case it is possible that the high priority traffic can overwhelm the scheduler and result in starvation of the lower priority traffic, i.e., the low priority traffic will never get served. This is an undesirable attribute of this type of scheduler and other schedulers will be investigated which do not have this attribute.

For our initial studies, we investigated the following active queue management scheme implemented on top of the two schedulers, i.e., strict priority and Deficit Round Robin (Keshav, 1997). Both queues maintain a counter reflecting the number of packets within their queues. Each queue is configured with a local threshold, i.e., $T_{high}^{(local)}$ and $T_{low}^{(local)}$ respectively, where $T_{high}^{(local)} \leq B_{high}$ and $T_{low}^{(local)} \leq B_{low}$. Here, B_{high} is the size (in terms of packets) of the high priority queue and B_{low} is the size (in terms of packets) of the low priority queue. Class $q = 0$ feeds the low priority queue and class $q = 1$ feeds the high priority queue. Further we model only two PLs, where $p = 1$ gets preferential treatment over $p = 0$. In the event that the buffer occupancy of the low priority queue equals or exceeds $T_{low}^{(local)}$, then the active queue management denies access to all packets with $p = 0$. In the event that the buffer occupancy of the high priority queue equals or exceeds $T_{high}^{(local)}$, then the active queue management denies access to all packets with $p = 0$. In the event that the buffer occupancies drop below their local thresholds, then their respective active queue management allows all PL traffic access to the queues [1].

These local thresholds help address the problem of preemption within each local queue, but there are cases where low QoS class data is tagged high PL and high QoS class voice is tagged low PL and we need to be able to communicate a low priority buffer overflow to the high priority buffer management in order to prevent PL inversion. To prevent this situation from happening, we implement one additional threshold in the low priority queue, i.e., $T_{low}^{(non-local)}$, where $T_{low}^{(local)} \leq T_{low}^{(non-local)} \leq B_{low}$. When the buffer occupancy of the low priority queue equals or exceeds $T_{low}^{(non-local)}$, then the high priority queue management scheme causes all low PL traffic to the high priority queue to be dropped. The activity remains in effect until the buffer occupancy in the low priority queue drops below $T_{low}^{(local)}$. When we set $T_{low}^{(local)} = T_{low}^{(non-local)} = B_{low}$ and $T_{high}^{(local)} = B_{high}$, then we effectively disable the active queue management and recover the standard two finite queue priority model.

The strict priority scheduler always checks the high priority queue and services all packet in queue prior to servicing a packet in the low priority queue. The Deficit Round Robin scheduler is described in (Keshav, 1997). Here each queue maintains a deficit counter. When the scheduler is ready to service the next packet it checks, in a round robin fashion the status of each queue. It first increments the queue's deficit counted by the amount specified by $Quantum_{High}$ or $Quantum_{Low}$ depending upon whether checking the high or low priority queue. If the size of the packet at the head of the queue exceeds the queue's deficit counter, then the packet is served and the queue's deficit counter is decremented by the size of the packet. Else, the scheduler moves to check the status of the other queue, incrementing its deficit counter and continuing as described.

[1] Clearly it is desirable to implement different upper and lower thresholds for this queue management scheme to prevent thrashing. However, for our initial studies and to simplify the initial analysis we implement this single threshold strategy. Later on we will implement the two threshold scheme to eliminate thrashing.

4. RESULTS

In this section we discuss our simulation results. Our simulation program logs all events to a log file. We have implemented a set of PERL scripts which analyze the log files and generate a set of metrics for the simulation output. The metrics we generate include the number of packets arrived and departed the system, the mean, variance and standard deviation of the system delay and service times, and the loss probability. These metrics are computed in aggregate and broken down according to the various QoS classes and Precedence Levels.

We first investigate performance of our PHB model under smooth traffic conditions for the strict priority scheduler. We define a voice and data stream pair according to the traffic models discussed in an earlier section. Our lowest offered load results are for a single pair of independent voice and data streams. For this single pair, we set their packets to carry $p = 1$. Then we increase the offered load by adding additional, identical arrival process pairs, except that the packets for the additional arrival processes are labeled with $p = 0$. For example, one set of runs consists of 6 arrival process pairs. There are six G.729a like voice streams and six 20 KBps data streams. One each of the voice and data streams are labeled $p = 1$ while the rest are labeled $p = 0$. The voice streams carry a QoS class marking of unity and the data streams carry a QoS class marking of zero. The overall offered load for this case is roughly 68.8% utilization of the DS-1 rate server.

We sized the buffers by running simulations until we found a buffer size that yielded a loss rate of about 1/1000 in the low priority queue. This resulted in a buffer size of 12 packets for low priority. We then (rather arbitrarily) set the size of the high priority buffer to 4. We then ran a series of simulation studies with these buffer sizes with and without the active queue management scheme enabled for increasing levels of offered load. Each simulation ran for 100 seconds of simulation time and handled over 10^5 packet departures.

In Figure 6 we show plots of the mean system delays per QoS Class, i.e., $q = 0$ or 1, and per Precedence-Level, $p = 0$ or 1. The upper plot shows the results when no active queue management is implemented. Here there is no discrimination between $p = 0$ or 1 traffic. We ran several scenarios ranging from an offered load of 10% to a high of roughly 115%. Because the data traffic is relatively smooth, the buffer sizes are small and the resulting delays are small as well. The high priority delays are significantly smaller than the low priority delays. The high precedence traffic is generated by one CBR flow and one data flow. The lower plot shows the corresponding system delays when the active queue management is configured. Here we set $T_{high}^{(local)} = 3$ packets, $T_{low}^{(local)} = 8$ packets, and $T_{low}^{(non-local)} = 10$ packets. We see little change in the delays for the voice streams. For the data streams we see a slight decrease in the delays. As well, we see that the system is now discriminating between the different PL traffic.

Figure 7 shows the loss results for the same set of simulation runs. The upper plot shows the results with no active queue management. Here we see packet losses in the low priority queue equally for both the PL 0 and 1 traffic. Clearly, this is not a desirable result. The lower plot shows the results when the CQAQM is configured. Here we see a slight increase in the loss probability for the voice and data streams tagged PL equal 0, while the loss probability for the voice and data streams tagged PL equal 1 is zero. This is the desired effect.

We now increase the burstiness in the traffic profiles. Here we set the Coefficient of Variation (Cv) for the data streams to 4, and repeat the process we followed above. We again made a set of simulation runs to size the buffers. This resulted in the low priority buffer size being set to 750 packets. We set $T_{high}^{(local)} = 3$ packets, $T_{low}^{(local)} = 500$ packets, and $T_{low}^{(non-local)} = 625$ packets. This large increase in the buffer size over the previous case is due to the increase in the burstiness of the arrival process.

Figure 8 shows the results for the system delay in the simulation runs. The upper plot shows the case where no active queue management exists. The lower plot show the comparable results with CQ-AQM configured. Of course, the actual delays have increased significantly over the previous simulation runs. However, the results are qualitatively similar. Turning on CQ-AQM slightly reduces the delays seen by the data packets. Figure 9 give the comparable loss results. Again, in the upper plot we see packet losses for both the PL equal to zero and one tagged traffic. When active queue management is configured the loss probability for all PL tagged traffic is zero. The PL zero tagged traffic shows a slight increase in their losses for both QoS class zero and one.

We also ran a set of studies where we modeled a Deficit Round Robin scheduler. A summary of all of our results, is found in Table 4. Here we present only the loss metrics for the various configurations for the case where the offered load was the maximum studied, i.e., an offered load of approximately 115%. For the case of the Deficit Round Robin scheduler, we ran two different sets of quantum sizes, i.e., $Quantum_{High} = 70$ and $Quantum_{Low} = 7$ or 70. We left the arrival processes and the buffer sizes and

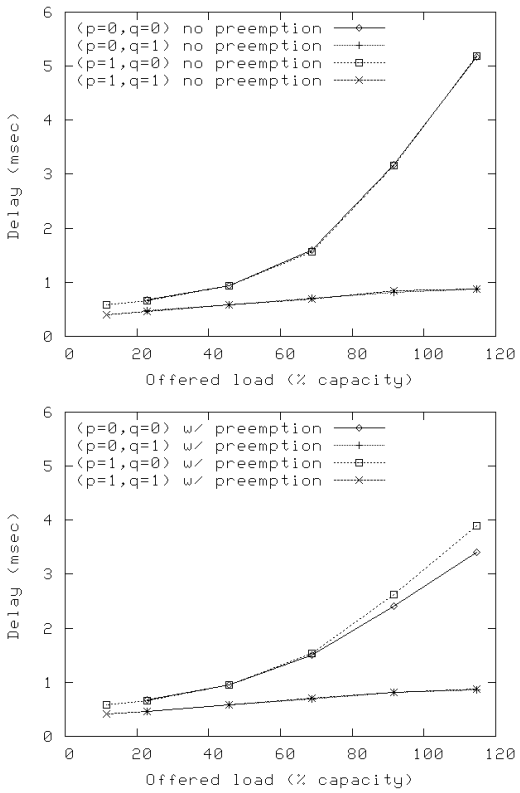

Figure 6: Example delay behavior without (upper) and with preemption (lower) for the Poisson data model.

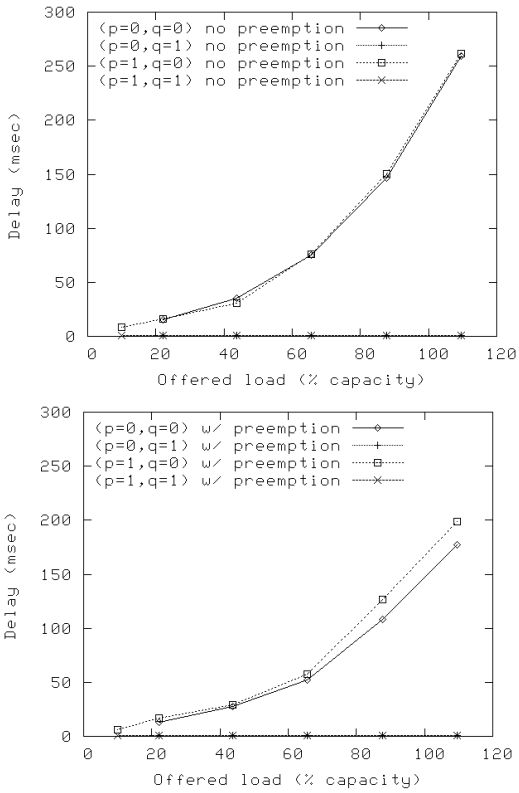

Figure 8: Packet delay without (upper) and with CQ-AQM (lower) for the data model with Cv = 4.

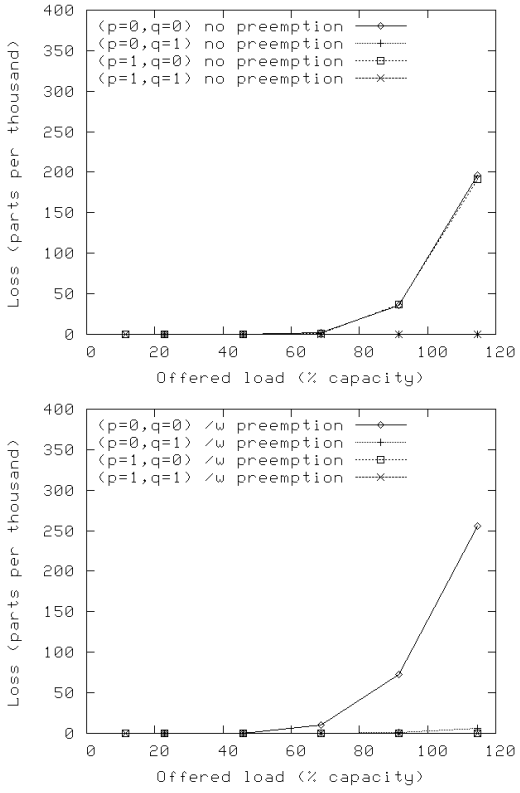

Figure 7: Example loss behavior without (upper) and with preemption (lower) for the Poisson data model.

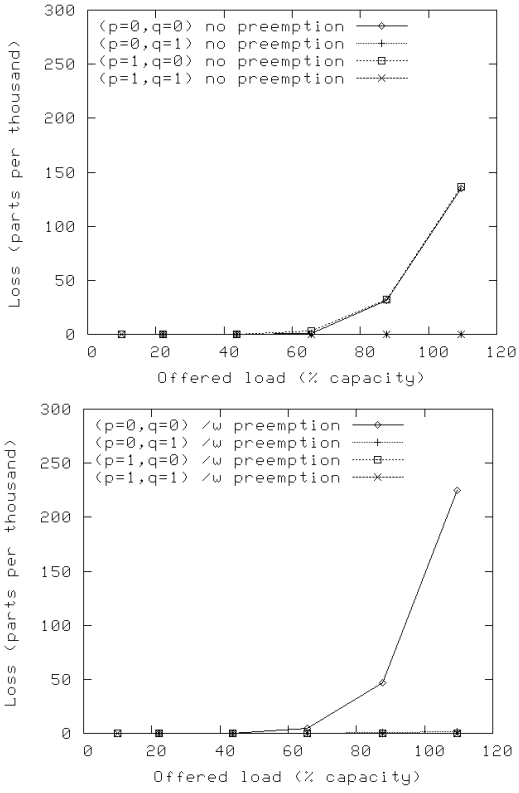

Figure 9: Packet loss behavior without (upper) and with CQ-AQM (lower) for the data model with Cv = 4.

Traffic Model	Scheduler	Losses (ppt) (1,1)	(1,0)	(0,1)	(0,0)
CBR+Poisson	Strict	0.4	191.0	0.1	196.0
		0	0	0	256.0
CBR+CV=4	Strict	0	136.2	0.2	135.3
		0	0	1.6	224.6
	Deficit	0	96.9	0.2	101.3
	(70&7)	0	0	1.4	144.6
	Deficit	0	106.4	0.1	106.6
	(70&70)	0	0	1.3	215.0
Internet	Deficit	3.5	237.5	5.8	229.3
	(70&7)	0.4	0	22.3	242.4

Table 3: Summary of loss results for the various cases simulated.

thresholds to be the same as for the previous strict priority scheduler and bursty traffic arrival processes. For each configuration listed in the table, the first row shows results for the case of no AQM and the second row shows the results for the CQ-AQM PHB. The results for the packet losses for the Deficit Round Robin cases are very similar to our previous simulation results; the CQ-AQM scheme is very effective in avoiding precedence inversion and protects the high precedence traffic regardless of QoS level.

Finally, we took our Deficit Round Robin scheduler and buffer configuration and investigated the impact of a more realistic arrival process on the performance of our CQ-AQM scheme. Here we activated ten independent traffic streams from our analysis of Internet traffic. One stream's traffic was tagged $(p = 1, q = 1)$, one stream's traffic was tagged $(p = 0, q = 1)$, one stream's traffic was tagged $(p = 1, q = 0)$, and the remaining seven streams' traffic were tagged $(p = 0, q = 0)$. We ran a set of studies where we used these traffic streams to generate offer loads of 20%, 40%, 60%, 80%, 100% and 120% of the server capacity. Again, we only show the loss results for the highest offered load results in Table 4. The results for the other load conditions were similar to previous studies at comparable loads. Even for this case, the CQ-AQM scheme is very effective in preventing Precedence Inversion. There is only a very small packet loss in the $(p = 1, q = 1)$ stream under CQ-AQM management. However, realize that we have simulated Internet-like traffic through a buffer configuration where the high priority buffer is only four packets deep. Typically, these buffers would be larger in reality.

5. CONCLUSIONS

These preliminary studies helped us to test our simulation model, and get some initial indication of how the CQ-AQM approach to precedence handling would work. The results are far from definitive, but are encouraging. We plan on more complete simulation studies in the future. Our focus will be to investigate the impact of other schedulers. traffic models and buffer arrangements.

ACKNOWLEDGMENTS

We are grateful for the numerous and enlightening discussions with the 2005 GIG Precedence and Preemption Team members.

References

B. Davie, A. Charny, J.C.R. Bennett, K. Benson, J.Y. LeBoudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. *An Expedited Forwarding PHB*. RFC 3246, IETF, 2002.

J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. The Assured Forwarding PHB group. RFC 2597, IETF, 1999.

S. Keshav. *An Engineering Approach to Computer Networking*. Addison-Wesley, 1997.

Leonard Kleinrock. *Queueing Systems Volume II*. John Wiley and Sons, 1976.

B. Liebowitz. An analysis of local behavior models for the military precedence and preemption transport. GIG Priority and Preemption Team, September 2005.

Chairman Joint Chiefs of Staff. The GIG requirements for precedence and preemption DRAFT. Technical Report CJCSI 6215.02a, Department of Defense, July 2004.

National Laboratory For Applied Network Research, http://mna.nlanr.net/ (Now taken over by CAIDA, the Cooperative Association for Internet Data Analysis, http://www.caida.org)

Kishor Trivedi, *Probability & Statistics with Reliability, Queueing, and Computer Science Applications* Prentice Hall, Inc. 1982