International Journal of Computer Applications Technology and Research
Volume 5– Issue 4, 218 - 225, 2016, ISSN:- 2319–8656

# A Survey of Packets Scheduling Congestion Control Algorithms in Internet Protocol Storage Area Networks

Joseph Kithinji
Department of Information Technology,
Meru University of Science and Technology
Meru, Kenya.

**Abstract**: Several approaches have been proposed to empower communication systems with quality of service (QoS) capabilities. In general, their main goal is to coherently support the end-to-end performance needs of applications, based on the establishment of, and agreement on, a set of concepts, policies and mechanisms. The fiber channel is the standard technology used in storage are network communications but it does not have mechanisms for providing QOS guarantees. However the increasing use of transmission control/internet protocol based network storage has introduced the possibility of using already existing techniques and tools to achieve to achieve QOS guarantees in Internet protocol Storage Area Networks. Due to the existence of other competing traffic in internet protocol networks it is necessary to provide storage input/output traffic with guaranteed network bandwidth. This paper discusses the available packet scheduling mechanisms pin pointing their advantages and disadvantages. The main goal is to combine two packets scheduling mechanisms and come up with a hybrid that assure a given storage QOS requirement between a storage client and internet protocol storage.
**Keywords**: Congestion control, packet scheduling, QOS, prioritization, fiber channel

## 1. INTRODUCTION

Computer networks were designed mainly to transfer data and email where QOS was effectively implemented by the use of TCP. As storage area networks become popular, many organizations networks have transformed into converged networks in which same infrastructure is shared to ensure all the requested services [20]. Although this convergence offers some advantages like sharing of network media, on the other hand it has come with some disadvantages. One being that it has led to the competition for network resources (buffers of routers), which leads to congestion [18]. Delay in delivering the packets, jitter, loss of packets are consequences of congestion. Since different applications show different sensitivity to these issues. For example, file transfer protocol is not affected by delay and jitter, whereas Storage area networks read requests are very sensitive to packet loss [9]. To solve this problem QOS was introduced to provide better Storage area networks performance.

Due to the increasing need for data storage and economy of scale savings, establishments for storage area networks has been increasing over the last years [19]. In the recent years the storage market has shifted from using expensive fiber channel technology towards TCP/IP based technology. Storage networking is adopting the TCP/IP networking which creates a need for providing QOS in order to offer guaranteed storage performance [14]. This is due to the fact that in TCP/IP networks there are other competing traffic for network bandwidth.

Guaranteed storage performance is an essential requirement for applications using it, and it's important for network designers to ensure that storage performance meets the requirements of the applications utilizing it. In a storage area network**,** a single host request may flood the resources of a storage pool causing poor performance of all hosts utilizing that particular pool [14]. Hence, the performance of a given host utilizing a shared pool resource is unpredictable by the nature of resource sharing. To address this problem a mechanism of providing QOS based on some policy is required. Storage service level agreements provide for predictability in service delivery which is not effective due to the absence of QOS mechanisms in storage devices [13]. However when we utilize TCP/IP as transport mechanisms, packet scheduling mechanisms which have been used and studied well are available.

Internet small computer interface is the technology used for implementing the IP-SAN [10]. The Internet Small Computer Systems Interface (iSCSI) is a TCP/IP – based protocol for establishing and managing connections between IP-based storage devices, hosts and clients. It defines the rules and processes to transmit and receive block storage applications over TCP/IP networks by encapsulating SCSI commands into TCP and transporting them over the network via IP. It is a protocol for new generation of data storage systems that natively use TCP/IP [14]. Since internet small computer system interface uses TCP for transportation, it is possible to use the available packet scheduling algorithms tools for the purpose of throttling traffic destined to storage devices.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 details the existing packet scheduling algorithms. Section 4 presents hybrid queues. Section 5 presents our discussion while section 6 concludes the paper.

## 2. BACKGROUND

Online data storage doubles every nine months due to the increasing demand for networked information services [13]. As a result network storage architectures have developed from network attached storage, to storage area network and most recently storage over IP (IP SAN). Internet SCSI (internet small computer system interface) is the technology used for implementing the IP-SAN. Providing QOS to Storage area networks has been a challenge which has led to the design of many approaches such as Stonehenge, cello, facode, triage, argon, chameleon and Aqua[16]. Despite all these research specification regarding QOS in Storage area networks utilizing TCP/IP have not been exhausted.HPLab storage systems department has been researching on how to provide Qos(based on response time and bandwidth)[16]. In this case bandwidth is to be allocated on demand.

Storage area networks technologies such as fiber channel and gigabit Ethernet have enabled storage systems to be maintained as storage pools, reducing the total cost of ownership[13,16]. Few protocols have been developed to support storage area network environment. Fiber channel protocol was developed for fiber channel based storage area networks. Internet small computer system interface was developed for internet protocol based storage area networks. A main advantage of internet small computer system interface is that it can operate on standard Ethernet; that is it can be able to exploit existing features and tools that have been developed for internet protocol networks [18, 19]. Thus this paper focuses on the storage environment using internet protocol where storage devices are attached to internet protocol networks and storage clients communicate with the storage devices via the internet small computer system interface.

A different storage client may require a different storage service called storage quality of service; that is each client requires receiving a guaranteed storage service independently of the status of the input/output services in other storage clients [8, 9, 11]. The use of internet protocol in storage area networks avails the commonly used QOS implementation techniques which can be applied in storage area networks.

The combination of well-known and trustworthy throttling mechanisms and an extended knowledge about storage systems internals makes an appealing pragmatic and non-intrusive approach to the problem of QOS in storage systems [17]. Instead of building new scheduling algorithms for storage devices, this paper suggests the utilization of previously known and trusted tools to implement QOS in storage area networks [6]. Since fiber channel does not provide prioritization, the use of transmission control/internet protocol as a transport mechanism makes the commonly used packet scheduling algorithms available.

The market for internet small computer system interface storage devices is growing making it an interesting target for quality of service research [8]. The integration of the well-known TCP/IP throttling mechanisms and storage systems internals provide a good approach to solving the problem of QOS in storage systems [7]. Adoption of existing systems would be more appropriate instead of introducing new algorithms which are bound to cause uncertainty and overhead.

# 3. PACKET SCHEDULING ALGORITHMS

## 3.1 FIFO Queuing

FIFO is the most widely used queuing discipline because of its easy configurations [2]. Packets belonging to different flows pick up in the FIFO queue and processed in the order of arrival.FIFO belongs to the unconscious group which treat packets as they are. Packets from all input flows are queued into a memory stack, then they are dequeued in the order of arrival one by one onto the output link. Since FIFO does not perform any reorganization of the queue, there is no schedule overhead experienced by packets [1]. This means turnaround time, waiting time and response FIFO time are low. However due to the lack of prioritization, FIFO systems may have trouble meeting deadlines.

On the other hand lack of prioritization ensures that every process will eventually complete its transmission without the risk of starvation [1]. All packets are placed in a single queue and are treated equally [4]. Packets are transmitted as bandwidth becomes available. Packets are treated accordingly to their arriving order. However since the queue buffer is finite, any packets that arrives after the buffer if full it is dropped without regard to which flow the packet belongs to or how essential a packet is [5]. This concept is known as the trail drop concept.

FIFO works well in links that are not heavily congested .Since works on first come first serve basis, if a node initiates a large file transfer, it can consume all the bandwidth link to the disadvantage of other traffic [18]. This phenomena is known as packet trains since the source sends a train of packets to its destination and packets from other hosts get caught behind the train. FIFO is efficient for large links that have little delay and minimal congestion [5]. To optimize QOS metrics such as buffer requirements and queue delay, FIFO uses traffic shaping techniques [6]. FIFO also employs AQM mechanisms to ensure fairness among flows.

## 3.2 Priority Queuing

Packets are assigned to classes which are associated with certain priority value. Packets with high priority are processed first. Priority queue is able to differentiate traffic hence reducing delay of important traffic. On the other hand if there is continuous flow of high priority traffic, low priority will be starved [6]. In basic implementations of priority queuing, it consists of four priority queues where packets are handled using FIFO. Packets belonging to the highest priority queue are serviced first. A packet scheduler is used to check for existence of packets in the highest priority queue after the current packet is processed. Any packets that arrive in the high priority queue are processed immediately [9]. The main advantage of priority queuing is that it is able to ensure highest priority to storage area networks(especially for read requests) but also lead to infinite delay for packets belonging to lower priority queues.

Priority queuing technique is suitable for situations where mission critical traffic needs preference [8]. Priority queue provides a smooth transition of important traffic (packets), through the network, using management at all intermediate points. Priority queuing classifies traffic with priority labels low, normal, medium and high. Packets which have not been assigned to a class automatically fall into the normal waiting queue. Data belonging to the high priority queue is handled first followed by that belonging to low priority queues. Priority queue currently uses static configuration and because of this it is not able to automatically adjust to the changing requirements in the network [14]. All incoming queues are assigned to a given network interface with each queue having a priority level.

When queues are being sent out the interface, they are scanned for packets in descending order of priority. The high priority queue is scanned first followed by the lower priority queues. The packet at the head of the highest queue is chosen for transmission [19]. This process is repeated every time a packet is to be sent [15]. Priority queuing has four traffic priorities; high, normal, medium and low (comparative study of different queue disciplines). Priority queuing is useful in environments where we want to make sure that mission critical traffic gets priority treatment. However priority queuing has a weakness in that it does not automatically adapt to changing network requirements due to static configurations [6]. Although priority queuing is simple, its implementation, it can cause queuing delay and increased jitter on the lower traffic.

## 3.3 Class based queuing

Priority queue is able to differentiate traffic hence reducing delay of important traffic. On the other hand if there is continuous flow of high priority traffic, low priority will be starved [6]. In basic implementations of priority queuing, it consists of four priority queues where packets are handled using FIFO. Packets belonging to the highest priority queue are serviced first. A packet scheduler is used to check for existence of packets in the highest priority queue after the current packet is processed. Any packets that arrive in the high priority queue are processed immediately [9]. The main advantage of priority queuing is that it is able to ensure highest priority to storage area networks(especially for read requests) but also lead to infinite delay for packets belonging to lower priority queues.

Priority queuing technique is suitable for situations where mission critical traffic needs preference [8]. Priority queue provides a smooth transition of important traffic (packets), through the network, using management at all intermediate points. Priority queuing classifies traffic with priority labels low, normal, medium and high. Packets which have not been assigned to a class automatically fall into the normal waiting queue. Data belonging to the high priority queue is handled first followed by that belonging to low priority queues. Priority queue currently uses static configuration and because of this it is not able to automatically adjust to the changing requirements in the network [14]. All incoming queues are assigned to a given network interface with each queue having a priority level.

When queues are being sent out the interface, they are scanned for packets in descending order of priority. The high priority queue is scanned first followed by the lower priority queues. The packet at the head of the highest queue is chosen for transmission [19]. This process is repeated every time a packet is to be sent [15]. Priority queuing has four traffic priorities; high, normal, medium and low (comparative study of different queue disciplines). Priority queuing is useful in environments where we want to make sure that mission critical traffic gets priority treatment. However priority queuing has a weakness in that it does not automatically adapt to changing network requirements due to static configurations [6]. Although priority queuing is simple, its implementation, it can cause queuing delay and increased jitter on the lower traffic.

## 3.4 Fair queuing

The fair queuing is a scheduling mechanism that classifies packets and processes these packets based on a service level agreement. Fair queuing uses a round robin algorithm to allocate bandwidth where every flow has an equal chance [6, 10]. The round robin algorithm ensures that flow from one class does not starve other classes off the bandwidth. The main advantage of priority queuing is that in a situation where there is congestion in a particular class, other classes are not affected and therefore the overall network performance is not affected. The downfall of priority queuing as a scheduling mechanism is that it does put into consideration the packet length. This means if a particular class has big flows, then the class may use more bandwidth and therefore take longer to be served [8]. However fair queuing is considered to be best suited in sharing bandwidth among different classes with the same bandwidth requirements.

## 3.5 Weighted Fair Queuing

In weighted fair queuing, incoming packets are grouped into classes and admitted to different queues[7]. Then these queues are assigned priority based on their weights, with high weights corresponding to high priority. Packets are then processed in a round robin manner with the number of packets selected from each queue based on the corresponding weight. For example in a case where we have weights 3,2 and 1,this would mean that three packets are processed from the first queue ,two from the second queue and one from the third queue. If the bandwidth manager does not impose priority on the classes, all weights can be equal. In this way we have fair queuing with priority [6]. All configurations in weighted fair queuing are automated with no room for tuning possibilities.

Queues from flows are grouped into a maximum of 256. The weighted fair queuing then uses the following notations:

SN=previous-SN (Weight*new packet length)…1

Weight =32384/ (IP-precedence+1)…..2

Where SN is the completion time [9].

Weighted fair queuing is mostly suitable in environments where it is desirable to provide a constant response time for the demanding users or applications without adding an excessive bandwidth. Weighted fair queuing implements bitwise fairness, which allows a queue to be served based on the number of bytes [9]. Weighted fair queuing ensures no traffic is starved off bandwidth. In this way low-level traffic can smoothly travel through the network. This increases service efficiency since an equal number of low-level and high level packets are transmitted. Weighted fair queuing can also automatically adapt to the changing network conditions. The weights are calculated from IP priority bits where values 0 to 5 are used (6 and 7 are reserved) and the weighted fair queuing algorithm calculates how many additional services must be provided for every queue [5]. Weighted fair queuing reduces the round trip delay which makes it perform better than TCP and in the process reducing congestion and speeding up slow connections.

Weighted fair queuing results in predictable behavior over the entire route while the response time for each active flow can be reduced by a multiple factor [2]. The weights are used also to determine how much bandwidth each flow is allocate relative to others, the maximum length of a queue is defined by the length limit [13]. Weighted fair queuing disciplines sorts packets in weighted order of arrival of the last bit to determine the transmission order. Transmission order bits can be used to identify weights. Weighted fair queuing is aware of packet sizes and can support variable sized packets, so that flows with large packets are not allocated more bandwidth than the queues with smaller packets [5]. Flows are grouped into those requiring huge amounts of bandwidth [6]. The goal is to always have bandwidth available for the low throughput flows to split the rest proportionally to their weights.

Since weighted fair queuing is derived from fair queuing, if N data flows are currently active with weights w1, w2…wn,

data flow number i then average data rate can be achieved using the equation below [13]

Average data rate=rwi/w1+w2+……wn.

By regulating the weights, automatically, weighted fair queuing can be used to achieve guaranteed data rate. Each flow is allocated with different bandwidth percentage hence preventing monopolization of the bandwidth by some flows.

## 3.6 Class Based Weighted Fair Queuing

Class based weighted fair queuing is based on the idea of weighted fair queuing, the only difference is that in class based weighted fair queuing, traffic flows are grouped in classes. The other difference is that classification is done manually in class based weighted fair queuing unlike in weighted fair queuing where configurations are done automatically [20]. This ensures flexibility in allocating a minimum bandwidth amount on the fair queuing basis as well as on the basis of administrator defined classes [2, 3]. If a traffic flow is not attached to any configured classes, it can use only the remaining link bandwidth which is not associated with any class. Each class is allocated a guarantee amount of bandwidth. Class based weighted fair queuing is used in situations where more low-priority flows could overflow the high priority data stream. However low latency queue can be marked so that actual high priority queue is differentiated. Traffic belonging to low latency queuing will be serviced before all other traffic placed in other classes and at the same time the necessary amount of bandwidth will be guaranteed [4]. When a class does not use all the guaranteed bandwidth, it can be shared among other classes.

## 3.7 Custom Queuing(CQ)

In CQ flows are categorized into 16 FIFO queues with a defined buffer length. Each of the FIFO queues is then assigned a suitable percentage of the total bandwidth. Scheduling of the queues in the output interface is done in round robin [2]. Although CQ can be able to eliminate the infinite delay experienced in priority queuing, it cannot implement priority for storage area networks. However a fine tuning of row lengths can help to reach acceptable results [9]. CQ guarantees mission critical flows a certain percentage of the whole bandwidth while assuring other traffic will get predictable through put [2]. For example if we have 16 queues, queue 0 is configured as a special queue called system queue which is used to handle keep alive and control packets that are considered as high priority packets. Queues 1 to 15 are used to carry user defined traffic. This means user traffic cannot pass through queue 0.

Traffic can be classified based on input interface access control lists, application types and packet sizes. Queues are then served in a round robin manner until a byte counter limit threshold is met. Once this threshold is met, the frame from the next queue are serviced [4]. In CQ routers service each queue sequentially transmitting a configurable percentage of traffic on each queue before moving to the next one. CQ routers determine how many bytes should be transmitted for each queue based on the interface speed and the configured percentages. When a particular queue is being processed, packets are sent until the number of sent bytes exceeds the byte count, or until the queue is empty [5].In this way all the traffic is processed.

## 3.8 Modified Weighted Round Robin

MWRR uses variable –sized packets to determine which queue to be served. To calculate the variable size, it uses a deficit counter variable to initialize to each queues weight. Before a queue is serviced, its deficit counter is initialized to the queues weight. A packet is scheduled for transmission if the deficit counter is greater than zero.

High priority packets are allowed to cut front of the line. The processed number of packets is equal to the normalized weight over the mean packet size [8, 11]. MWRR queuing discipline serves packets at the head of every non empty queue whose modified counter is greater than the size of the packet at the head of the queue.

## 3.9 Deficit Weighted Round Robin(DWRR)

DWRR was proposed by M.Shreedher and Varghese in 1995 [20]. DWRR handles packets of variable size without any consideration of the mean size. The number of the packet size is subtracted from the packet length and packets that exceed the packet number are held back until the next visit of the scheduler. DWRR uses scholachastic fair queuing to assign data flows to queues [17]. Queues are served in a round robin with a quantum of service attached to each queue [16]. This implies that if a queue is unable to send packets due the size, the remainder from the previous quantum is added to the quantum for the next round.

Queues not serviced in a round are compensated in the next round. However once a flow is serviced it must wait for N-1 other flows to be serviced before it is serviced again. During each round, a flow transmits its entire quantum data once as a result DRR has poor delay [7]. Each queue is associated with a quantum and a deficit counter. Quantum represents the number of bytes that each queue can send on its turn [3]. The deficit counter variable is used to keep track of the credit each queue possesses for sending traffic and is initialized to zero.

## 3.10 Modified Deficit Round Robin(MDRR)

In MDRR when a queue is served a fixed amount of data in bytes is dequeued. The algorithm then services the next queue. If the amount of data dequeued exceeds the value configured, in the next round less data will be dequeued to compensate for the excess data that was previously dequeued. As a result, the average amount of data dequeued per queue will be close to the configured value [3, 4]. Queues in MDRR are defined using two variables; a quantum value (number of bytes served in each round) and a deficit counter (used to track how many bytes a queue has transmitted in each round).

Packets are served only when their deficit counter is greater than zero. Each packet served decreases the deficit counter by a value equal to its length in bytes [6]. Queues with deficit counters as zero or negative are not served. In each new round the deficit counter of each non-empty queue is increased by its counter.

# 4. HYBRID WAITING QUEUES

Because different queuing mechanisms have different advantages, the idea is to combine different queuing mechanisms and join their positive but also negative properties into new hybrid queuing method. The main

disadvantage of hybrid queuing is the duplication of the memory of the mechanism which forms a queue [19]. This is due to the fact that every memory element and its size involve certain latency or delay for traffic which goes through these interfaces. The higher the number of these interfaces the bigger the delay which is detrimental to applications such as Storage area networks read requests [3, 5, 8]. This is why we have to make a compromise between the number, size and length of the intermediate buffers to avoid excessive data spillage (or data loss). When a buffer is too small and to also avoid scenarios where the buffers are too big, and are increasing the delay. This aspect is called the jitter effect.

### 4.1 The custom queuing –class based weighted fair queuing hybrid waiting queue

Combine CQ and class based weighted fair queuing. In the first phase CQ is used to allocate bandwidth among all active applications to avoid congestion. Once packets are sent to the output CQ interface they arrive to the class based weighted fair queuing input interface [20]. Class based weighted fair queuing arranges traffic into classes defined by a class-based weighted fair queuing algorithmically the advantages of the class based weighted fair queuing are retained[3]. With this method we reduce the delays within the network, which is not the case with ordinary CQ scheme**.**

### 4.2 Priority queuing-class based weighted fair queuing hybrid waiting queue

In the first step priority queuing is used to arrange flows into waiting queues according to the priorities set in individual packets TOS fields[6]. The output interface of priority queuing algorithm first serves the highest priority data streams and then all other lower ranking queues. As packets leave the priority queuing interface they have already been assigned bandwidth as configured by the network administrator. This way the packets at the class based weighted fair queuing mechanism output interface do not need to fight for bandwidth as it is guaranteed in advance [5]. This accelerates the transfer of high priority flows and such flows become independent of all other lower priority flows.

The priority queuing-class based weighted fair queuing is closely related to low latency queues. The low latency queuing mechanism allows a class that is served as a strict priority queue. Traffic in such class will be served before all other traffic in the remaining classes. Bandwidth amount is also guaranteed in this case [3]. This implies that all traffic which is above the level of bandwidth reservation is simply discarded

### 4.3 Weighted fair queuing-class based weighted fair queuing hybrid waiting queue.

Weighted fair queuing is implemented in the first step to ensure fairness for all applications since internal weighted fair queuing are emptied by principle of fairness. Weighted fair queuing ensures that undisturbed flow throughput for all active applications. In the next step the class based weighted fair queuing puts packets into admin defined classes. This way every application at the first stage gets a fair treatment and in the second phase high priority applications gets its own classes with the pre-reserved bandwidth. The rest of the bandwidth is left for all other active applications [6, 7]. The fairness and fluidity movement apply for all active applications.

Weighted fair queuing is effective for operating with IP priority settings such as resource reservation protocol. Weighted fair queuing is also capable of managing round trip delay problems. This is the main reason for combining the weighted fair queuing and class based weighted fair queuing [20]. The weighted fair queuing-class based weighted fair queuing is at the same time capable of accelerating slow features and removing congestion in the network. The results become more predictable over the whole routing path, while Ethernet delays can be greatly decreased compared to CQ, priority queuing and weighted fair queuing [3]. The weighted fair queuing and the class based weighted fair queuing combination can represent the best solution for reducing the Ethernet delay.

## 5. DISCUSSION

In this paper we have looked at packet scheduling algorithms that can be used to reduce congestion in storage area networks. We have looked at the strengths and weaknesses of each algorithm. First we looked at FIFO which is considered a simple mechanism since when there is no congestion packets do not experience any delays. On the other hand if there is congestion packets will experience delays due as a result of queuing delay. Therefore FIFO cannot be used to provide quality of service guarantees. This makes FIFO a very weak scheduling technique due to the fact that internet traffic is always bursty which in all cases leads to congestion. This weakness of FIFO makes it no to be implemented alone.

We have also looked at priority queuing which has got an advantage in that in a situation where there is congestion in a particular class, other classes are not affected and therefore the overall network performance is not affected. This ensures that a particular class misusing its bandwidth does not affect the other classes.

The downfall of priority queuing as a scheduling mechanism is that it does put into consideration the packet length. This means if a particular class has big flows, then the class may use more bandwidth and therefore take longer to be served. This may end up starving other classes and does not allow bandwidth sharing. Priority Queuing is most appropriate where we have a fixed number of queues requiring different priorities. But due to the fact that the high priority traffic is transmitted first, priority queuing cannot be used to offer end to end service guarantees. However priority queuing can be combined with leaky bucket algorithm to ensure that high priority queues do not monopolize the link. However fair queuing is considered to be best suited in sharing bandwidth among different classes with the same bandwidth requirements. This is because with fair queuing every class gets a guaranteed fair share of the bandwidth with allowance for sharing if not in use. Fair queuing ensures that all traffic has fair access to network resources. This prevents bandwidth starvation for less aggressive traffic.

Weighted fair queuing attaches weights to every flow which determines the amount of bandwidth associated with a particular flow hence guaranteeing a constant rate of bandwidth. Configurations in weighted fair queuing are do automatically which makes possible to provide QOS guarantees since it is able to adapt to changing network conditions. Class based weighted fair queuing improves on the weighted queuing by classifying flows so as to offer differentiated service. Configurations are also done manually which provides more flexibility in assigning bandwidth to traffic flows. Flexibility in configurations allows for network administrators to vary the configurations now and then which leads to a more customized bandwidth allocation for a particular organization.

Since different queuing mechanisms have got different advantages, they can be combined into hybrid queues to offer better quality of service guarantees. CQ can be combined with class based weighted fair queuing which would result in reduction delays associated with CQ. Priority queuing and class based weighted fair queuing can be combined to provide bandwidth guarantees not available in priority queuing. The combination of queuing disciplines in congestion control may result in improved results however there is added overhead due to the processing required by the particular combined mechanisms.

## 6. CONCLUSIONS

In this paper we have discussed packet scheduling techniques that can be used to achieve quality of service in storage area networks. We have discussed each technique separately in order to get insight on its strengths and weaknesses. From our discussion it is evident that no single technique can achieve quality of service guarantees in storage area networks. Hybrid approaches may result in better quality of service guarantees but, results in jitter affect which is caused by duplication of the memory mechanisms where the queues are formed. This is because every memory element and its size involve a certain latency for traffic which goes through that interface.

In future research we aim at exploring techniques of reducing the jitter effect caused by combining scheduling mechanisms.

## 7. ACKNOWLEDGMENTS

My thanks goes to Professor Muchiri Muketha for his advice in the development of the paper.

**Table 1.Comparative analysis of packet scheduling algorithms in providing QOS**

| **MEASURE OF QOS** | **FIFO** | **priority** | **fair queuing** | **weighted fair queuing** | **custom queuing** | **Modified Weighted Round Robin** | **Deficit Weighted Round Robin** | **The custom queuing – class based weighted fair queuing hybrid waiting queue** | **Priority queuing-class based weighted fair queuing hybrid waiting queue** | **Weighted fair queuing-class based weighted fair queuing hybrid waiting queue.** |
|---|---|---|---|---|---|---|---|---|---|---|
| starvation | High | High | Low | Low | Very low | Medium | Medium | Low | Low | Low |
| Packet loss | High | High | Very low | Very low | Low | Low | High | Medium | Medium | Low |
| Bandwidth guarantee | Low | Low | Medium | High | High | Low | Low | High | High | High |
| delay | High | Very high | Medium | Low | Low | Medium | Medium | Low | Low | Low |
| Jitter | High | High | Low | Low | Low | Low | Low | Very low | Very low | Very Low |

## 8.REFERENCES

[1] Harpreet, K, Gurpal & S, Fatehgarh. (2011)" Wimax Networks Implementation and Evaluation of Scheduling Algorithms in Point-to-Multipoint Mode", IJCST Vol. 2, Issue 3, India.

[2] Ahmad, K & Bahauddin, Z (2011)."VoIP Performance Over different service Classes under Various Scheduling Techniques", Australian Journal of Basic and Applied Sciences, 5(11): 1416-1422-CC, ISSN 1991-8178.Pakistan.

[3] Sasa K, Amor C, Joze M & Zarko C.(2012)","Influences of Classical and Hybrid Queuing Mechanisms on VoIP's QoS Properties" .University of Maribor.

[4]Rajeev,S, Sukhjit & S, Sumeet.(2015),"International Journal of Advanced Research in Computer and Communication Engineering "Vol. 4, Issue 3,india.

[5] Sarhan M. Musa, Mahamadou T, Matthew N. O. ,Pamela O & Roy G. P.(2013)."A Comparative Study of Different Queuing Scheduling Disciplines", Journal of Engineering Research and Applications Vol. 3, Issue 6, pp.1587-1591, Malaysia.

[6] Nidhi, M & Anil, K.(2013)."Active Scheduling (Queuing) Algorithms in Congestion Management: A Review ", International Journal of Digital Application & Contemporary research Volume 1, Issue 8, India.

[7] Ahmed K. (2011)"VOIP performance over different service classes under various scheduling techniques", Australian journal of basic and applied sciences (11):1416-1422, Pakistan.

[8] Sadafale, K, Barahate & Prashant, J. (2010),"Improvement and analysis of voice data traffic in VOIP", International conference and workshop on emerging trends in technology (IC WET).ACM, New York.

[9] Szabolcs, S. (2013)."Analysis of the algorithms for congestion management in computer networks", Carpathian Journal of Electronic and Computer Engineering 6/1 3-7 3 ISSN 1844 – 9689, Hungary.

[10]Sulbha, S, Hemke, V, Gawande, D & Gautum,L.(2013)."ISCSI-the future of storage Network ", international journal of application or innovation in engineering and management Volume 2, issue 4,ISSN 2319-4847,Australia.

[11] Rajesh K & Vinay.(2012)."Performance Evaluation of Scheduling Algorithms in WLAN Network with CBR Application using Qualnet", International Journal of Electrical, Electronics and Computer Engineering 1(1): pp1-5 ISSN No: 2277-2626,India.

[12] Dhaini, R,Assi, C.M,& Shami,A.(2008),"Dynamic bandwidth allocation schemes in hybrid TDM/WDM passive optical networks",IEEE Consumer Communications and Networking Conference.Vol.1,no.7,Germany.

[13] Prasad, R, Murray, M, Dovrolis, C, & Claffy, K. (2011) Bandwidth Estimation: Metrics, Measurement Techniques, and Tools, IEEE.Vol. 17, no. 6(May), pp. 27-35,Scotland.

[14] Devajit, M, Majidul A & Utpal, J.B. (2013)A study of Bandwidth Management in Computer Networks, International Journal of Innovative Technology and Exploring Engineering (IJITEE) .Vol.2, no.2,pp 20-30,Australia.

[15] Farhangi, S & Rostami, S.(2013),"A Comparative Study Between Combination of PQ and MWRR Queuing Techniques in Ip Network Based on OPNET", Middle-East Journal of Scientific Research 13 (8): 1051-1056, 2013 ISSN 1990-9233, IDOSI Publications,Turkey.

[16] Borgeengen, J, Haugerud, H.(2010),"Using traffic shaping to achieve ISCSI service predictability", 24th Large installation system administration conference.USENIX association, Norway.

[17]Van der Stok,P, D. Jarnikov,D,Kozlov, S,van Hartskamp,M & Lukkien,J.J(2009),"Hierarchical Resource Allocation for Robust In-Home Video Streaming", Journal of Systems and Software.Vol. 80, no. 7, pp. 951–961,Ireland.

[18]Van Rensburg, J.R, Veldsman, A, & Jenkins, M. (2008).From technologists to social enterprise Developers, ICT for development practitioners in Southern Africa.Vol.14, no.1, pp 76–89,Australia.

[19]Xu, X, Liang, D, Jiang,H & Lin,X.(2009)Dynamic Bandwidth Allocation in Fixed BWA Systems,Proceedings of the International Conference on Communication Technology (ICCT), vol.2,no.6, pp.1000-1003,Shangai.

[20]Yi-Nung, L, Meng-Che, C, & Shao-Yi, C.(2009)Bandwidth and local memory reduction of video encoders using Bit Plane Partitioning Memory Management,IEEE International Symposium on .Vol.14, no.8, pp.766-769, 24-27,Shangai.