RC22757 (W0303-121) March 24, 2003
Computer Science

IBM Research Report

A Case for Service Differentiation in SANs

Khalil S. Amiri, Kang-Won Lee
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

A Case for Service Differentiation in SANs

Abstract

Motivated by the need to simplify data sharing and curb rising storage management costs, storage systems are becoming increasingly consolidated. This is in turn leading to large and complex storage area networks connecting hundreds of storage devices and servers. Database, mail, multimedia and file servers, for example, gain access to a shared pool of storage devices through a common storage area network. The traffic carried over such storage networks is therefore increasinly heterogeneous, both in terms of its characteristics as well as its utility and its relative importance and value to the organization. Contrary to widespread misconceptions, storage network bandwidth is not overly abundant, and many deployed networks are reported to be oversubscribed. In this paper, we argue that storage area networks should provide differentiated services to storage traffic to ensure that the performance of business critical applications is not unduly harmed by low-priority batch traffic such as backup and long running data mining applications. In particular, we discuss the challenges of providing differentiated services in storage networks and, building on previous work addressing a similar problem in wide-area networks, we present an architecture that achieves such differentiation in practice. Furthermore, we propose a mechanism to ensure proper bandwidth allocation in the particular case of Fibre Channel Arbitrated Loop, which is the most widely deployed Fibre Channel network topology.

1 Introduction

Motivated by the need to enable easier data sharing and curb rising storage management costs, storage systems are becoming increasingly consolidated and thereby shared by a large number of users and applications. This consolidation is enabled by the emergence of high-bandwidth cost-effective storage area networks (SAN), such as Fibre Channel [4] and Gigabit Ethernet [6]. In traditional direct-attached systems, storage devices are attached to hosts using busses supporting a limited number of devices. Such systems therefore have limited size and are easier to understand, monitor, and manage. Emerging SANs, on the contrary, are full-fledged complex networks, including potentially thousands of devices, and sometimes stretching outside the "machine room" to remote locations across the campus or the metropolitan area.

The recent trend has been towards larger and more complex storage networks. This trend is likely to persist for several reasons. First, organizations are promoting the centralization of resources under a single authority responsible for storage management and administration across the organization. Centralization of storage resources, traditionally scattered behind the servers in various departments, into a single network, reduces management costs and makes sharing easier and more efficient. Other reasons driving the growth of storage area networks include the plummeting cost of storage devices and the explosive growth in application storage requirements.

As SANs increase in size and complexity, the traffic they carry becomes quite heterogeneous. Backup, online transaction processing, multimedia streaming, and decision support applications must all actively share and compete for the bandwidth available in the network. Given the inherent difference in their performance requirements as well as the relative importance of such applications to the organization, users often desire the storage network to take such high-level information into consideration when allocating network resources for competing traffic. Indeed, quality of service support in storage area networks has been cited as a pressing requirement in practice [10].

At first glance, it might appear that current storage networks should or can be designed to have ample bandwidth. On careful examination, however, we find that most storage area networks are in fact oversubscribed. That is, contention for network bandwidth is in fact a frequent occurrence. There are

several reasons for such contention. First it is simply cost-prohibitive to build a storage network that can carry the peak bandwidth from thousands of storage devices. Current storage devices can reach effective transfer rates of 15-20 MBps. Therefore, a small number of devices can easily saturate a 100 MBps Fibre Channel link. Furthermore, designing networks to have sufficient bandwidth for all circumstances is intrinsically difficult as workloads and device data rates change. Moreover, empirical evidence suggests that application bandwidth and throughput requirements are only roughly taken into account during network design. From these observations, we conclude that there will be circumstances when the bandwidth resources are constrained and thus have to be judiciously allocated to applications, according to high-level policy goals.

In this paper, we argue that there is a pressing need for service differentiation in storage networks. In particular, and contrary to widespread misconceptions, we argue that bandwidth on storage networks is limited in most practical deployments. We describe a variety of storage traffic types and their bandwidth requirements to further elucidate the need for service differentiation. We then present a high-level architecture to implement proper service differentiation in complex storage area networks. Furthermore, we present a mechanism to ensure proper bandwidth allocation in Fibre Channel Arbitrated Loop, the most widely deployed configuration of Fibre Channel networks.

The rest of the paper is organized as follows. Section 2 presents background on storage networks and storage traffic. Section 3 presents a general architecture for achieving differentiation in SANs. Section 4 proposes a mechanism to implement proper bandwidth allocation in a distributed fashion on a Fibre Channel Arbitrated Loop. Section 5 briefly reviews related work, and Section 6 concludes the paper.

2 Background

In this section, we provide some background on the various technologies used to implement storage area networks. Then, we describe the characteristics of the storage traffic generated by a set of representative application classes. We also highlight the disparate bandwidth and latency requirements typical of these applications.

2.1 Storage area network technologies

Storage area networks refer to networks used to connect storage devices or subsystems (e.g. disk arrays) to storage clients. Storage clients are host machines that directly access the back-end storage devices over the network. Such host machines include file, database, or multimedia servers.

We do not purport to describe all the SAN interconnect technologies in this section. Instead, we present the mostly widely deployed technology, namely Fibre Channel, and its emerging contender, Gigabit Ethernet. By far, Fibre Channel is currently the most widely deployed storage area network technology. Fibre Channel defines the physical, link, network, and transport layers for storage area networking. It also supports a notion of different service classes depending on the types of guarantees provided in terms of channel capacity and delivery guarantee. Fibre Channel defines three different interconnection topologies: point to point, arbitrated loop, and switched fabric. Point-to-point links are used to connect two devices. Arbitrated loop allows the connection of a larger number of devices (up to 127). Devices often have dual loop attachments allowing them to participate in two Arbitrated Loops, for fault-tolerance and increased bandwidth. Switched fabrics allow the switching of multiple point-to-point links or loops. They allow the interconnection of multiple nodes and loops into a switched configuration. Consequently, they can be used to build a larger network of devices.

Gigabit Ethernet, although not yet widely deployed, is likely to gain in acceptance with wider implementation of the iSCSI standard [1], which allows storage traffic to be encapsulated over TCP connections. In addition to the benefit of using commodity network adapters and the de facto standard TCP/IP protocol, using TCP/IP over Gigabit Ethernet introduces further flexibility in SAN design by allowing remote access to the storage system over wide-area IP networks.

2.2 Storage traffic

Storage area networks carry traffic on behalf of a variety of applications. Table 1 describes some typical application classes and their bandwidth and latency requirements.

As the table shows, applications have various traffic requirements and high-level service goals. For instance, OLTP requires predictable a high-throughput service to satisfy response time requirements of on-line users. Backup tasks can potentially consume large amounts of bandwidth and complete quickly, or extend for a longer period of time, consuming smaller amounts of bandwidth. On the other hand, restoring databases or filesystems from backups is usually associated with a higher priority than backup tasks.

Bandwidth intensive applications such as data mining, network log analysis, large file transfers, are often batch applications running in the background, and

	Request size	Read/Write ratio	Throughput (IOs/sec)	Bandwidth (MB/s)	Typical service requirement
DB (OLTP)	$\sim$ 4 KB	3	high	low	Guaranteed predictable throughput
DB (DSS)	$\sim$ 32 KB	8	low	high	DSS Query: Finish by specified approx. time
Streaming	64 – 256 KB	high	high	high	Guaranteed predictable bandwidth
Backup/restore	large	N/A	low	low	Backup: finish task by specified approx. time
Backup/restore					Restore: complete as soon as possible
Instructional	4 – 8 KB	5.6	low	low	Best-effort
Research	4 – 8 KB	3.7	low	low	Best-effort
NT office use	4 – 8 KB	6.3	low	low	Best-effort
Collaborative	4 – 8 KB		low	low	Best-effort
Webmail server	4 – 8 KB	2 – 3	low	low	Best effort/Average throughput
Web groupware	4 – 8 KB	2 – 3	low	low	Best effort/Average throughput

Table 1: Workload characteristics of key applications and typical service requirement that might be associated with their storage traffic. The table shows average request size, read/write ratio, and throughput and bandwidth requirements. The last column specifies high-level service requirement goals for the different traffic classes corresponding to these applications. Traffic characteristics are based on our own synthesis of measurements and previous studies. The instructional workload refers to the file system accesses of general purpose workstations. Research refers to the file system access of workstations used for research and development by computer science researchers. Collaborative applications consist of groupware applications such as Lotus Notes. Webmail is a typical web email service. Web groupware are sites supporting virtual communities. OLTP stands for online transaction processing workloads. DSS stands for decision support systems.

should not be allowed to unduly interfere with on-line workloads. However, some data intensive applications, such as decision support queries may sometimes be important, as a user may be waiting for such queries to complete to take important decisions.

We argue that proper classification of storage traffic, either based on the source application, or on a finer grained basis, is important and worthwhile. Classes of traffic are then associated with service level goals, and are guaranteed a given delay or bandwidth, based on their task completion time requirements and their relative degree of importance. Such classification scheme may be an iterative process involving administrators monitoring end application performance and changing bandwidth allocations. While this is an important topic of investigation, it is beyond the scope of this paper. For the rest of the discussion, we assume that traffic has been properly classified, with each class either being associated with an explicit delay or bandwidth requirement or declared as best-effort.

3 *Quality of storage* framework

This section presents a general framework for providing differentiated services in a storage area network. This paper refers to storage quality of service succinctly as *quality of storage*. We consider a distributed storage system, as shown in Figure 1, consisting of hosts connected through a storage area network to storage devices. A host, or storage client, issues read and write requests to the storage devices. Requests initiated by hosts are intercepted by storage caches, placed at the edge of the storage area network, which may serve them locally. All the requests that miss in the cache are sent over the storage network to the target storage device.

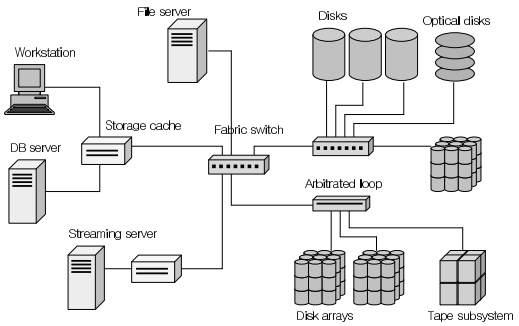

Figure 1: Storage area network.

As mentioned in Section 2, the vast majority of SANs use Fibre Channel as the underlying network. The SCSI storage access protocol is encapsulated in Fibre Channel frames and sent using Fibre Channel as the transport. If the underlying network implements the IP protocol, SCSI command and data transfer messages are sent over TCP connections, which are in turn mapped onto the FC transport. In this discussion, we focus on storage area networks implementing SCSI on top of Fibre Channel, although the bulk of the discussion applies to all storage area network technologies.

3.1 *Quality of Storage*: The model

We assume that I/O requests and response are tagged with the class of service they belong to. This can be in practice achieved in various ways. Requests may be tagged based on the originating application or machine, or based on the target logical unit (LUN) they

are accessing. We define a simple model for quality of service, allowing applications to express their requirements in terms of bandwidth or average access latency. We denote by $C = \{c_1, c_2, \ldots, c_n\}$ the set of application classes.

From the workload analysis of various storage applications, we identify two important service classes for Quality of Storage: bandwidth-intensive applications (e.g., streaming) and delay-sensitive applications (e.g., OLTP). To support the first type of applications, we define a service level goal that specifies a long-term bandwidth guarantee. In particular, the storage network can reserve resources along the path from the host to the device to guarantee a minimal available bandwidth. To be precise, the service goal can be stated as follows: "the average effective bandwidth for class c_i must be greater than equal to b_i^* measured over T_b." Here b_i^* represents the target capacity available to class c_i, and T_b represents a measurement time window.

The second service model of the QoS scheme is to satisfy a given average access latency for disk I/O operations measured over a certain time interval. More specifically, the service model can be described as follows: "the average access latency of class c_i must be less than equal to l_i^* (msec) measured over T_m (min)." Here l_i^* represents the target access latency of class c_i, and T_m represents a measurement time window. T_m is typically in the order of a few tens of minutes or a few hours.

In the shared storage architecture described above, the storage access latency l_i of class c_i is mainly determined by the following four parameters: cache access latency (l_{local}), hit rate in the proxy cache (h_i), network latency to access the remote storage location (l_{remote}), and disk access latency (d_i).

$$l_i = h_i * l_{local} + (1 - h_i) * l_{remote} + d_i. \quad (1)$$

Assuming that l_{local} is a small constant, the access latency l_i is effectively determined by the hit ratio h_i, remote storage latency l_{remote}, the disk access latency d_i. Thus, Eq. 1 can be rewritten as:

$$l_i = \epsilon + (1 - h_i) * l_{remote} + d_i. \quad (2)$$

Note that each of the terms in the above equation is directly related to the service offered by each component: hit ratio h_i is determined by the cache, remote latency l_{remote} by the network, and disk access latency d_i by disk scheduling. Indeed, mechanisms at the various components of the storage architecture are required to enforce QoS. In other words, it is possible to control the access latency l_i of class c_i by specifying a proper QoS level at the cache, and network components, and the disk. Note also that it is possible to achieve a limited degree of service differentiation by controlling one of the component if other components do not provide any service differentiation and hence in a long term offer an equivalent service to all the classes.

Among the three components, our focus in this paper is service differentiation at the storage network level. Mechanisms to enable service differentiation at storage cache and QoS-supporting disk scheduling algorithms have been extensively studied in the literature (for example, [8] for cache QoS and [9] for disk QoS; see Section 5 for overview).

Network delay can be generally decomposed into two parts: (1) propagation delay and (2) queueing delay. Since propagation delay is fixed and relatively small in SANs, we focus on service differentiation in queueing delay. Note that queueing delay is a function of the bandwidth reserved on the network path (with respect to the bandwidth requirement of the application). Thus, we can simply transform the delay service model to a bandwidth service model for most practical purposes. We therefore focus on providing a guaranteed bandwidth service for aggregate storage flows in a storage area network.

3.2 Policy-based management of *Quality of storage*

The entire quality of storage architecture is managed by an entity called the *policy manager*. The policy manager is responsible for specifying and enforcing a set of rules to be executed under certain system conditions. For instance, all service level violations or predicted violations in the near future detected by the policy manager will generate corresponding actions to correct the situation.

To achieve this goal, the policy manager communicates with various entities that perform admission control, resource reservation, traffic analysis, resource provisioning, and policy enforcement. The admission control and resource reservation modules ensure that the contracted QoS can be supported by the current infrastructure and dedicate the required amount of resources. The traffic analyzer communicates with the policy manager and provides application traffic characterization and performance predictions of the entire SAN. Based on the analysis of the traffic analyzer and the policy rules specified by the administrator, the resource provisioning module may generate actions to acquire more resources (by dynamically allocating resources from a "free pool" or by invoking procurement requests) before service contracts get actually violated.

The policy manager assumes the role of global coordination amongst the active components of the storage system to achieve Quality of Storage.

3.3 Providing guaranteed bandwidth service

We define a guaranteed bandwidth service as the core service of a storage area network providing Quality of Storage.

At network level, guaranteed service can be defined by borrowing the concept of Internet QoS such as integrated services. The integrated services (intserv) architecture defines three levels of service: guaranteed service, predicted service, and best-effort to each individual network flow. The guaranteed service provides an absolute performance guarantee of packet delivery from one end to the other by dedicating maximum required network resources to the path of the flow. The predicted service provides a statistical assurance on the end-to-end performance by reserving enough amount to ensure that the flow doesn't observe network congestion. The best-effort service does not provide any service guarantee.

In the target environment, the storage traffic will consist of a mix of long-term traffics (e.g., MM streaming or backup) and short bursty traffics (e.g., file access). Therefore we propose to classify the applications into a few intserv classes based on the workload characteristics by aggregating short-term traffics, and instantiate intserv mechanisms to provide guaranteed bandwidth service. To achieve this goal, we can adopt the network scheduling algorithm (e.g., WFQ, WRR) at the fabric switches. In addition, existing technologies proposed for the Internet QoS can be readily employed in our architecture. For example, RSVP (reservation protocol) can be used for resource reservation along the path, bandwidth broker can be employed for admission test, and congestion manager can effect congestion control of aggregated I/O connections.

Note that enabling intserv like network QoS will be seamless in a SAN based on Gigabit Ethernet interconnection. For those based on Fibre Channel can still adopt most of the network scheduling algorithms in the Fibre Channel switch logics. However, implementing an equivalent functionality in the Fibre Channel Arbitrated Loop is not a straightforward task. The next section further investigates details on this issue to complete the picture.

4 Bandwidth allocation in FC-AL

Fibre Channel Arbitrated Loop (FC-AL) is a shared Gigabit transport. Up to 127 devices can be attached to a single loop. While FC-AL is a shared medium just like Ethernet and Token Ring, it differs from them in that it is not a broadcast medium. Messages transmitted from a source to a target device are not visible to all devices in the loop. As a result, arbitrating for the loop employs a different protocol than the carrier sensing approach of Ethernet for example. In this section, we describe the Aribtrated Loop architecture and medium access control protocol, then present a scheme for bandwidth allocation in such a network.

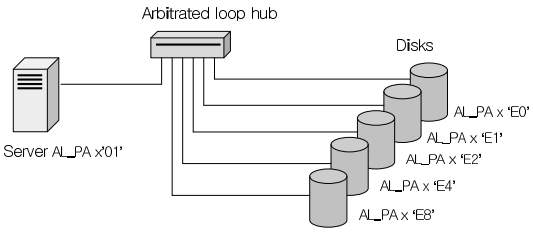

Figure 2: Physical wiring of a loop using a hub.

4.1 FC-AL protocol

Devices in FC-AL are attached using point to point links, with each device having a transmit and receive port. The transmit port of the last device in the loop is connected to the receive port of the first, thereby forming a loop. Data is sent downstream through the transmit port. Responses also flow in the same direction around the loop and are received at a device on the receive port. Devices receiving frames not addressed to them are relayed from the receive port to the transmit port until they reach their final destination. In order that a failure of a single device does not break the entire loop, this logical loop topology is usually implemented often using a hub as shown in Figure 2. When a device is unavailable, the hub bypasses it, thereby short circuiting it out of the loop. When a source device performs a transaction with a target device, the data sent downstream from source to target device is visible only to the devices that are downstream from the source device. The other devices will not see the data but will forward the acknowledgement from the target back to the source.

Each device on the loop has a unique address, called physical address, or `AL_PA`. Before beginning a transaction, a device must arbitrate and gain ownership of the

loop. Gaining control of the loop involves an arbitration step. Winning this arbitration step is governed by two factors, the device's physical address (AL_PA), and whether the device has recently acquired the loop. In the presence of multiple devices requesting the loop, the device with the highest priority address wins arbitration. To ensure fairness, a device that has won abitration and completed its transaction is required to back-off from requesting the loop until the loop transitions to an idle state (all currently interested devices have used the loop).

The arbitration process proceeds as follows. When a device needs access to the loop, it transmits an arbitration primitive, a four byte message containing two special bytes followed by the device's address in the last two bytes, ''k28.5 D20.4 AL_PA AL_PA'' [5]. This is referred to as ARB(x) message for short, where x is the device's address (AL_PA). This arbitration message can be sent even if another device currently owns the loop. However, in that case, this message is transmitted only between frames. When a device upstream receives the ARB(x) message, it fowards it along, unless it is also interested in gaining access to the loop. In that case, the device compares the address of the requesting device to its own. If its address has higher priority than the device originating the request, it can then substitute the ARB(x) message for its own. When a device receives its own ARB(x) message, it means that the message has made a full circle around the loop, and dominated the concurrent requests for the loop by any other devices. At this stage, the device wins arbitration and can begin its transaction.

Fairness is controlled as follows. When a device wins arbitration of the loop, its sets a local register, called the access flag to 0. The device does not attempt to arbitrate for the loop again unless this access flag is set. This flag is set when the loop is detected to be in an idle state, that is all devices interested in gaining access to the loop during this round have done so. Specifically, when a device has control of the loop, it substitutes any ARB(x) requests it receives by ARB(F0) and passes them downstream. ARB(F0) is dominated by all ARB(x) requests, so a device receiving this message that is still interested in the loop will substitute ARB(F0) for its own primitiave ARB(x). However, the current loop owner keeps substituting this low priority ARB(F0) to ensure that no arbitration primitive makes it full circle around the loop, and hence no other device will win arbitration, until the current transaction is completed. When the current owner completes its transaction, it forwards unmodified any ARB(x) primitives, allowing the next owner to be selected. When the loop is idle, a current filler word (IDLE) is sent around the loop, instead of ARB(x) or ARB(F0). Upon detection of this IDLE word, a device that has previously used the loop can set its access flag, and is free to arbitrate for the loop once more.

4.2 Bandwidth allocation in FC-AL

FC-AL ensures that bandwidth is equally shared by all nodes on the loop. While devices with higher-priority addresses will win arbitration in a specific round, the fairness protocol ensures that they refrain from requesting the loop in the subsequent rounds until all other interested devices have been allowed a chance to use the loop. This effectively results in dividing the bandwidth evently across active devices. For example, 20 active devices on a loop with a 1 GB/sec available bandwidth will be observe an effective bandwidh of about 50 MB/sec ($=\frac{1024\ MB}{20}$).

We assume that a high-level policy associates flows with a specific class of service. A flow can be thought of as a long-term connection, such as a SCSI session, set-up between a host and a given logical unit (LUN). A flow, f_{ij}, associated with a LUN hosted on device D_j, can either be a best-effort class, or a guaranteed service class with a given bandwidth, B_{ij}. Of course, the bandwidth requested by all the ative flows at any time should be lower than the effective loop bandwidth, B_{EL}, which is the raw loop bandwidth minus the various protocol overheads:

$$\sum_{j \in (1..N)} \sum_{i \in (1..F_j)} B_{ij} < B_{EL}$$

In the above equation, N is the number of devices on the loop, and F_j represents the number of flows ending at device D_j. Note that the above constraint is ensured by the admission control policy, which is implemented in a central policy manager. Once admitted, a guaranteed service class is assured that there is sufficient loop bandwidth to meet its target bandwidth requirement. To enforce this guaranteed bandwidth, we desire a mechanism to ensure that all admitted classes achieve their long-term bandwidth targets, B_{ij}. Furthermore, we would like excess bandwidth in the loop to be allocated to best-effort traffic fairly across the loop. Note that the standard loop arbitration protocol cannot be used to meet uneven bandwidth allocation across flows, given that it is designed to ensure fair bandwidth allocation across active devices on the loop.

To clarify the presentation, we first present a simple and inefficient protocol, the explicit rate scheme, then discuss its shortcomings and evolve it into a more efficient protocol.

Explicit rate control. The first scheme we present relies on each device ensuring that its guaranteed traffic

is queued to the network FC interface at a rate that is equal to the bandwidth allocated to that class. Such control can be implemented in a software layer above the FC protocol stack. That is, a class f_{ij} at device D_j which is allowed bandwidth b_{ij} is allowed to queue messages to the interface only at the long-term rate of b_{ij}. In the absence of best-effort traffic, this local device flow traffic policing is sufficient to ensure proper bandwidth allocation.

Excess bandwidth not used by the guaranteed service classes should be allocated to best-effort traffic around the loop. A device has to ensure however, that it does not inject "too much" best-effort traffic in the network because that will reduce the effective bandwidth allocated to guaranteed service classes. A scheme is required to ensure the amount of best-effort traffic injected in the network does not surpass the available "excess bandwidth", not allocated to guaranteed service classes. One scheme is for the policy manager to divide the excess bandwidth equally across the devices:

$$B_{BE} = B_{EL} - \sum_{j \in (1..N)} \sum_{i \in (1..F_j)} B_{ij}$$

Each device around the loop is assigned B_{BE}/N, and is allowed to send best-effort traffic only at that specified rate. The above scheme can be implemented without any modification to the low-level Fibre Channel protocol layers. The device can implement such policing outside the loop transport protocol. However, such a scheme has several disadvantages. First, it does not use resources efficiently. Consider the case when a single device around the loop has best-effort traffic to send. The device would be limited to the a fraction of the available excess bandwidth. The scheme is also not dynamic, in that it does not respond well to the availability of "free bandwidth." Ideally, available bandwidth should be allocated to any potential requester. For example, if a a single flow is active, it should be allocated the entire loop bandwidth.

Bimodal arbitration scheme. To address the disadvantages of the above scheme, we propose a mechanism which can use resources more efficiently, by properly exploiting excess bandwidth while ensuring that guaranteed service classes achieve their long-term bandwidth targets. This scheme associates two outgoing message queues at each device, a best-effort queue (BEQ) and a guaranteed service queue (GSQ). Data read from disk and ready to be sent back to a host is queued to the BEQ, regardless of which class it belongs to. Messages belonging to a guaranteed service class are moved from the BEQ to the GSQ according to the rate allocated to the corresponding class. Effectively, this will control the rate at which the guaranteed service queue is populated, such that it approximates the rate at which each class should be sending data. Messages in the GSQ have higher priority to use the loop, and only when the GSQ is empty are messages dequeued from the BEQ for sending. Furthermore, the device uses a different protocol to arbirate for the loop based on what kind of message it is sending (BEQ or GSQ). A device requesting the loop on behalf of a GSQ message always wins arbitration against a device requesting the loop on behalf of best-effort traffic. Only when no GSQ messages are contending for the loop, does a device win arbitration on behalf of a BEQ message.

Dual model arbitration proceeds as follows. A device inspects first the GSQ message list, processing outgoing messages from that list until it is empty, only when it is empty does the device inspect the BEQ message list and dequeues a message from there for sending. If the device has a GSQ message to send, it requests the loop using a GS arbitration primitive, specified as the following four byte message, `K28.5 D20.4 GS AL_PA`. On the other hand, if it has a BSQ message to send, it requests the loop using the arbitration primitive `K28.5 D20.4 BE AL_PA`. The GS byte is chosen such that it dominated the BE byte. Therefore a GSQ arbitration primitve dominates all BE arbitration primtives. If two devices request the loop using two arbitration primitives of the same type (both GS or both BE), the winner is determined by the originating device's address as in the basic protocol.

This protocol is essentially a two-mode protocol, consisting of a GS mode and a BE mode. The loop is considered to be in the GS mode, whenever any device is requesting the loop using the GS-mode arbitration primitive. When in a GS mode, only guaranteed traffic can compete for loop bandwidth. When no device has GS data to send (all GSQ's are empty), devices can win control of the bus to send best-effort data. Note that if a guaranteed service class requires more bandwidth than the minimum guaranteed to it, and the loop has available free bandwidth, the class will be able to use this excess bandwidth. A device can send the messages for that class when they are still in the BEQ and before they are moved to the GSQ.

Guaranteeing fairness in this dual mode scheme is interesting. In the BE mode, fairness can be guaranteed in a simple manner using the access flag, as in the basic FC scheme. However, guaranteeing fairness in the GS mode is more involved. Note that if the device were to reset the access flag after completing a GS-mode transaction, and wait until the loop is idle, this might yield the loop to best-effort traffic. The scheme should allow other GS-mode devices to grab control of the loop, but not devices operating in the

best-effort mode. We solve this problem by introducing two arbitration primitives in the GS mode, `K28.5 D20.4 GS1 AL_PA` and `K28.5 D20.4 GS2 AL_PA`, such that GS1 dominates GS2. The GS1 primitive is used initially to request access to the loop. When arbitration is won, and if the device still has data to send, it uses the GS2 primitive to arbitrate again for the loop. Consequently, the device will lose arbitration to other devices using the GS1 primitive, thereby giving a chance to other devices to use the loop. The device would cycle between using GS1 and GS2 primitives for arbitration. Note that GS1 and GS2 both dominate the BE byte, therefore devices in the GS mode will always win arbitration over best-effort traffic. [1]

5 Related Work

Service differentiation has been investigated thoroughly in the context of the wide area Internet. The integrated services (intserv) and differentiated services (diffserv) represent two approaches which can be applied to the problem of storage differentiation in storage area networks. We believe that approach based on the IntServ model is more appropriate for SANs because a single centralized network resource allocation and management authority can be envisioned in such centalized environment under a single administrative domain.

There is a rich body of literature on bandwidth allocation and service differentation in computer networks. In particular, many researchers have investigated bandwidth allocatin in shared broadcast networks, such as Ethernet [3, 12]. Fibre Channel however is not a true broadcast medium, and therefore requires special protcols for bandwidth allocation. Similarly, the real-time community has investigated providing guaranteed service in networks. In particular, mechanisms have been proposed to ensure that real-time synchronous traffic in Token Ring networks can meet its required deadlines [2]. Sychronous and periodic messages initated by source devices are ensured to reach their destination devices before their specified deadlines. These scheme also divide the bandwidth of the ring across competing synchronous traffic, by having each device limit the amount of data it sends in a given rotation of the token. While storage traffic is not usually associated with hard real-time deadlines, some of this work is related and can be used in developing quality of storage architectures for SANs.

[1] Note also that although this scheme still favors a high priority address device to win the loop again after switching from GS2 to GS1, this is not a big problem since outgoing rate control at the devices ensures that long-term bandwidth is properly allocated. This scheme ensures, however, that in the short-term, waiting times to gain control of the loop are reduced.

Service differentiation at the level of individual storage devices or subsystems has also been studied quite extensively [7, 9]. In [9], Shenoy et al. proposed the Cello disk scheduling architecture that has two levels of disk scheduling structure: (1) the class-independent scheduler governing the coarse grain allocation of bandwidth, and (2) the class-specific scheduler that controls the fine-grain interleaving of requests. Cello implements three types of class-specific controllers for interactive best-effort, throughput-intensive best effort, and real time applications. In [11], Wijayarantne et al. presented a disk scheduling architecture to support multiple classes of service using admission control and a proper disk scheduling. In particular, the proposed an algorithm to provide deterministic guarantees to VBR (variable bit rate) applications through statistical multiplexing.

6 Conclusions

In this paper, we observe that storage networks are increasing in size and complexity, and are therefore carrying increasingly heterogeneous traffic. We argue therefore that service differentiation in storage networks is a pressing and critical requirement. We describe storage traffic and its bandwidth requirements, and present an architecture to implement proper service differentiation in complex storage area netowrks. In particular, we present a scheme to ensure proper bandwidth allocation in Fibre Channel Arbitrated Loop, the most widely deployed configuration of Fibre Channel network.

References

[1] IETF IP Storage (IPS) Charter. On-line document, `http://www.ietf.org/html.charters/ips-charter.html`, August 2002.

[2] G. Agrawal, B. Chen, W. Zhao, and S. Davari. Guaranteeing synchronous message deadlines with the timed token protocol. In *International Conference on Distributed Computing Systems*, pages 468–475, 1992.

[3] P. Anelli and G. L. Grand. Differentiated services over shared media. *Lecture Notes in Computer Science*, 2092, 2001.

[4] CERN. http://hsi.web.cern.ch/HSI/fcs/spec.htm.

[5] T. Clark. *Designing Storage Area Networks.* Addison Wesley, 1st edition, 1999.

[6] Gigabit Ethernet Alliance. http://www.gigabit-ethernet.org/.

[7] R. Golding, E. Shriver, T. Sullivan, and J. Wilkes. Attibute-managed storage. In *Workshop on Modeling and Specification of I/O (MSIO)*, October 1995.

[8] B. Ko, K. Lee, K. Amiri, and S. Calo. Scalable Service Differentiation in a Shared Storage Cache. In *The 23rd International Conference on Distributed Computing Systems*, June 2003.

[9] P. J. Shenoy and H. M. Vin. Cello: A disk scheduling framework for next generation operating systems. Technical Report CS-TR-97-27, 1, 1998.

[10] Storage Admin. Quality of Service for SANs. On-line document, `http://www.storageadmin.com/Articles/-Index.cfm?ArticleID=26283`, August 2002.

[11] R. Wijayaratne and A. L. N. Reddy. Providing QOS guarantees for disk I/O. *ACM Multimedia Systems Journal*, 8(1), January 2000.

[12] Yoshigoe and Christensen. RATE Control for Bandwidth Allocated Services in IEEE 802.3 Ethernet. http://citeseer.nj.nec.com/471682.html.