Tertiary Disk: Large Scale Distributed Storage

Nisha Talagala, Satoshi Asami, Tom Anderson, David Patterson
{nisha, asami, tea, pattrsn} @cs.berkeley.edu
University of California at Berkeley
May 1997

Abstract

In the past 5 years, disk costs have been falling at a factor of 2 per year. Today, terabyte capacity disk storage systems are feasible. Given the rapidly increasing areal density and disk transfer rates, these systems will have significant cost/performance advantages over tape libraries of similar capacity. If commodity hardware is used, large disk systems can avoid the high cost of custom designed disk arrays, as well as their limitations on scalability. This paper presents Tertiary Disk, a 3TB disk storage system built from commodity hardware. Tertiary Disk uses PCs and switched networks to connect 370 8GB disks. We show that even though commodity hardware is used, the overall system can be more reliable than a single disk. A cost analysis of our prototype shows that the additional infrastructure needed to create a terabyte scale storage system is a fraction of the cost of the underlying disks. In comparison, the costs of large disk arrays are many times the cost of the underlying disks. We also present performance measurements from our prototype, and show that the PC architecture is a good match for hosting a large number of disks. Overall, we show that storage systems designs like Tertiary Disk have cost/performance and reliability advantages over most choices available today for terabyte scale storage.

1. Introduction

In the past 5 years, the cost performance gap between secondary and tertiary storage has been widening. The cost per megabyte of disk drives has been falling at a factor of 2 per year, compared to 1.5 per year for tape drives and libraries. Disk areal densities have been increasing at 60% per year, with 8 GB 3.5 inch disk units shipping by late 1996. Data rates have also been increasing at rates of 40% per year, expected to pass 40 MB/s by the end of the decade [1]. These trends change the possibilities in large scale storage systems. If they continue, large storage systems composed of disks will have significant cost/performance advantages over tape libraries of similar capacity.

Applications such as databases, video on demand, medical data and web archival have a need for storage systems which are high performance as well as high capacity [2,3 4]. The solution used in most cases is a hierarchy of a disk array and tape library. However, disk arrays have drawbacks in terms of cost/performance, availability, and scalability. Due to custom hardware, the cost per megabyte of RAID disk arrays increases with system capacity, unlike raw disks and tape systems. Also, a disk array needs to be connected to a host computer, which becomes a bottleneck for both performance and availability. Its scalability is limited by the number of disks that can be supported by the infrastructure. Some storage consuming applications like web archival have a fixed growth rate of data. When such applications reach the capacity limit of their disk array, another array must be added. Adding independent disk arrays also lowers the reliability of the total system and complicates storage management.

In this paper we present Tertiary Disk, a storage system architecture which exploits the trends mentioned above to create large disk storage systems that avoid the disadvantages of custom built disk arrays. The name comes from twin goals: to have the cost per megabyte and capacity of tape libraries and the performance of magnetic disks. We use commodity, off the shelf components to develop a scalable, low cost, terabyte capacity disk system. Tertiary Disk uses PCs connected by a switched network to host a large number of disks. Our prototype consists of 20 200MHz Pentium Pros, which host 370 8GB disks. The Pentium Pros are connected through a switched network of 160 MB/s Myrinet links. In the following sections we discuss the hardware and software architecture and present some preliminary performance measurements. Section 2 compares our prototype to commercial disk arrays and tape libraries. Section 3 describes the hardware and software architecture. Section 4 shows that even though Tertiary Disk uses a large number of independent components, the overall system can have a mean time to failure greater than a single disk. Section 5 analyzes the cost of our prototype and Section 6 gives preliminary measurements. Section 7 uses the performance measurements to discuss the trade-offs between configurations. Section 8 describes related work in distributed storage systems and Section 9 concludes.

2. Motivation

The previous section suggested that current trends in storage media make terabyte scale disk systems feasible. In this section we show the motivation for building such systems out of commodity components. We compare Tertiary Disk to various tape libraries and disk arrays in the capacity range between 200GB and 5TB. Since the systems being compared are so different from one another, in both media and target workload, we compare them on two metrics, their cost/capacity and peak bandwidth. For tape libraries we assumed the peak bandwidth to be the total peak bandwidth of all drives in the system. Note that this is optimistic as other things, like tape mounting times, also affect performance. For the disk arrays the peak bandwidth is assumed to be the bandwidth of the link(s) to the host. Figures 1(a) and 1(b) show where some commercial tape and disk systems fall in $/MB and bandwidth. The performance numbers for Tertiary Disk are based on measurements of single unit prototypes which will be discussed in following sections.

As can be seen from figure 1(a), Tertiary Disk is competitive in cost with disk arrays and in capacity with tape libraries. Our investigation suggests that tape libraries have improved in cost/megabyte by 25% per year for the last several years, while disks have improved at 100% per year over the same period. If these trends continue, Tertiary Disk systems can be cheaper than tape libraries in 2-4 years. While the systems compared have different features which to some extent affect their cost, they have some trends in common. The cost of similar capacity tape systems differs by the number of drives available and the technology of the drive (ex. 8mm vs. DLT), but in general the cost/Megabyte of tape systems decreases with increasing capacity. The cost of disk arrays differs mostly with the available memory and the bandwidth of the host connection. For example, the Sun RSM and EMC 3300 differ in price because the former has 256 MB of memory and 80MB/s host connection while the latter has 2GB of memory and a 320MB/s host connection. However, disk arrays in general show an increasing cost/MB with capacity. This increase is due to the cost of custom designing a larger system.

The main point is that as Tertiary Disk is based on smaller commodity systems connected through a network, it is able to scale in bandwidth while keeping cost/capacity relatively constant. Scaling can also be done with disk arrays, for instance, by replacing a large disk array with some number of smaller arrays. But in this case each array has to be connected to a host, with additional software to coordinate the hosts. This approach has two main disadvantages. First, the connection between the host and the disk array is much smaller than the aggregate bandwidth of the underlying disks. Second, if all the independent disk arrays are to appear as one large storage system, additional software will be needed to coordinate layout of data between the hosts. In this case the custom features provided by the disk array controller may become useless. What we are proposing is to avoid the host to disk array connection, connect the disks directly to a host, and use software to coordinate the hosts. Such a design is able to fit in between the tape and disk arrays in cost, and scale better than either in performance.

It is difficult to do a fair comparison of the reliabilities of these different systems, specially as tape is a different media. However, the manufacturer quoted reliability for tape

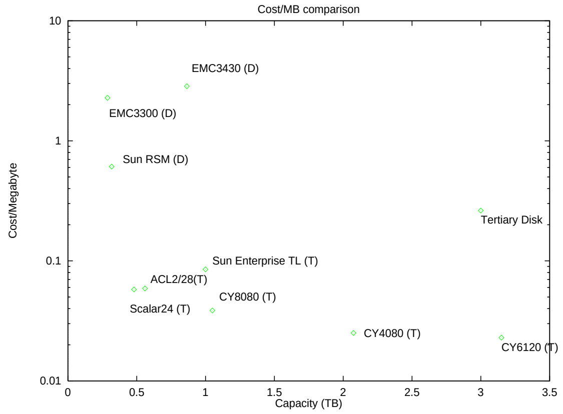

Figure 1(a): Comparison on the basis of cost/capacity

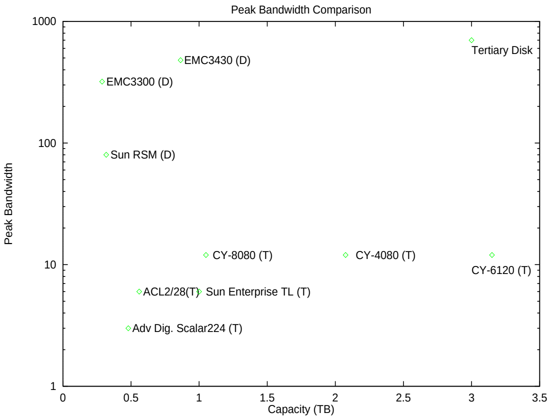

Figure 1(b): Comparison on the basis of aggregate peak bandwidth of all links to hosts.

systems is around 50,000 hours. Disk arrays typically quote higher reliability (in millions of hours), however this is usually assuming the disks are configured in a parity group and not accounting for failures in the host. In section 4 we do an evaluation of the reliability of Tertiary Disk and show that it is possible to develop a terabyte disk system which has a higher mean time to failure than a single disk.

3. Architecture and Implementation

The Tertiary Disk prototype that we are constructing consists of 20 Pentium Pros which host 370 8GB disks. The PCs are interconnected through a switched network of Myrinet links. The PC cluster is also connected through Myrinet to a cluster of 100 UltraSPARCS. Both clusters will be running xFS, a serverless distributed filesystem. This section describes the architecture and implementation of the hardware. Since detailed discussion of the software is beyond the scope of this paper, we give a brief overview of xFS and provide references for more detailed information. [6,7]

To study the trade-offs between different hardware configurations, our design is based on two designs for individual *nodes*. A single Tertiary Disk node is composed of two PCs which share disks. This *double ending* of disks to two PCs is for higher reliability. If a PC has a hardware or software failure, all disks connected to it are accessible through its dual PC at the other end of the string. Figures 2(a) and 2(b) show the logical design of the two nodes. From now on these two designs will be called Node design 1 and Node design 2. Both nodes use the Fast-Wide SCSI disk interface. The SCSI strings are shared between PCs, with two SCSI controllers per string. In normal mode (i.e. with no failures), each PC accesses half the disks.

Both node designs have four SCSI controllers per PC. In Node design 1, each SCSI string has 8 disks in one disk enclosure. In the second node design, each SCSI string has 14 disks in two disk enclosures. In both node designs, each PC has one Myrinet network interface. The complete prototype consists of eight nodes of design 1 and two nodes of design 2. In sections 5 and 6 we compare the cost and performance of the two node designs.

Hardware:

PCs are a natural choice for hosting disks. The main system bus has a peak bandwidth of 132 MB/s, and it is possible to connect an arbitrary number of devices using PCI-PCI bridges. Our PCs have enough expansion slots for four additional expansion cards on the main system bus. We use these slots for 1 Myrinet interface card and up to 3 dual channel SCSI controller cards. Since a single wide SCSI string allows 16 devices and all SCSI strings have two controllers, each string can host a maximum of 14 disks. Therefore this configuration can have a maximum of 84 disks per node. More details on the implementation of double ending can be found in [5]. If PCI extension boxes are used, this number can be increased arbitrarily. Our PCs have 64MB of memory each and are running Solaris 2.5.1. In addition to the Myrinet network, the machines are connected through switched ethernet.

Power and cooling for the disks is provided by the disk enclosures. For easier maintenance and monitoring, the enclosures are hot pluggable and programmable from a remote host through a serial port. These features are important as in large storage systems, management can be as expensive as the storage itself. All of the components of two nodes of design 1, or one node of design 2, fit in one 19-inch wide by 7 foot tall rack. Each rack contains PCs, disk enclosures and network switch hardware.

Software:

Figure 3 shows the integration of the PC cluster with the Ul-

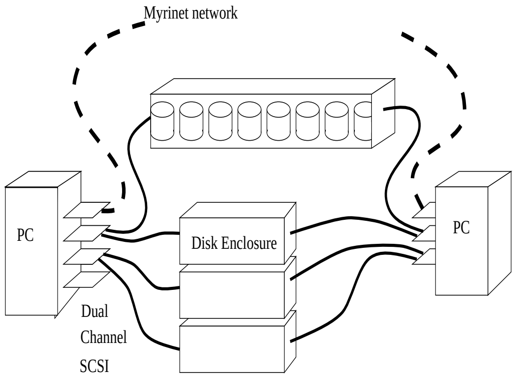

Figure 2(a): Logical design of Node 1

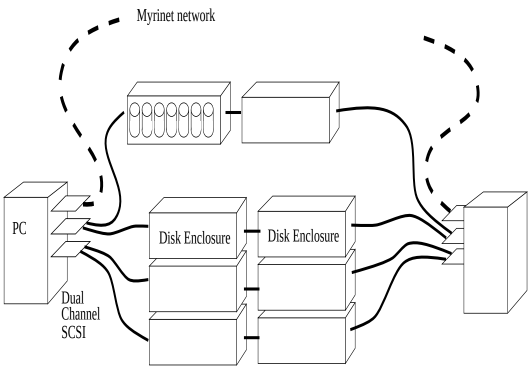

Figure 2(b): Logical Design of Node 2

traSPARC cluster. For simplicity all Tertiary Disk nodes are shown alike, although some actually have more disks than others. Both clusters will be running xFS. More detailed descriptions of xFS can be found at [6, 7]. xFS is composed of three main modules, the client, manager and storage server modules. All applications run above the client module, and all file system services are provided by the client and manager modules. The disks are managed by the storage server modules. As shown in Figure 3, the UltraSPARCS run the client and manager modules, and the PCs run only the storage server modules. xFS is a log structured file system. The files used by applications are laid out by the client modules onto stripe groups, where each stripe group contains disks from multiple storage servers. The storage servers write data in a log structured form on their own disks. Figure 3 shows a stripe group that spans all 10 Tertiary Disk nodes. Each node contributes 1 disk to the stripe group. If this is a RAID 5 stripe group, the parity is calculated by the client module. This minimizes the communication between the storage server modules and simplifies their design. It also allows the number of storage servers to increase without additional complications. Through xFS's use of stripe groups across storage servers, any client can stripe data across multiple PCs so that a single PC is no longer the bottleneck for access to the disk bandwidth. This helps improve reliability, as parity groups need not be local to a single PC. Also, the scalable nature of xFS helps the scalability of Tertiary Disk.

4. Reliability

The previous section showed how a terabyte capacity disk system can be designed from commodity hardware. But would such a system be reliable? The problem when putting large numbers of systems together to build a larger system is that the reliability is inversely propositional of the number of independent components. Tertiary Disk uses double ending and network striping to improve reliability. Double ending allows all disks in a single node to be accessible even after the failure of a PC or a SCSI controller. Reliability of data across nodes can be improved by defining parity groups which are orthogonal to the nodes. In this section we do a general failure analysis of Tertiary Disk using the principles of data redundancy provided by the RAID work [8,9,10,11]. Since different layouts of stripe groups are possible, we evaluate the example configuration given in Figure 3.

The types of failures we consider are disk failures, PC failures, SCSI controller failures and SCSI string failures. We do not consider disk enclosure failures, as our disk enclosures have redundant power supplies. We also do not consider failures in the network infrastructure. There are two reasons for this. First, a thorough analysis of different network topologies and their failure characteristics is beyond the scope of this paper. Second, for any given network topology, a similar capacity storage system made up of disk arrays will face similar concerns. The goal of this section is to show that the addi-

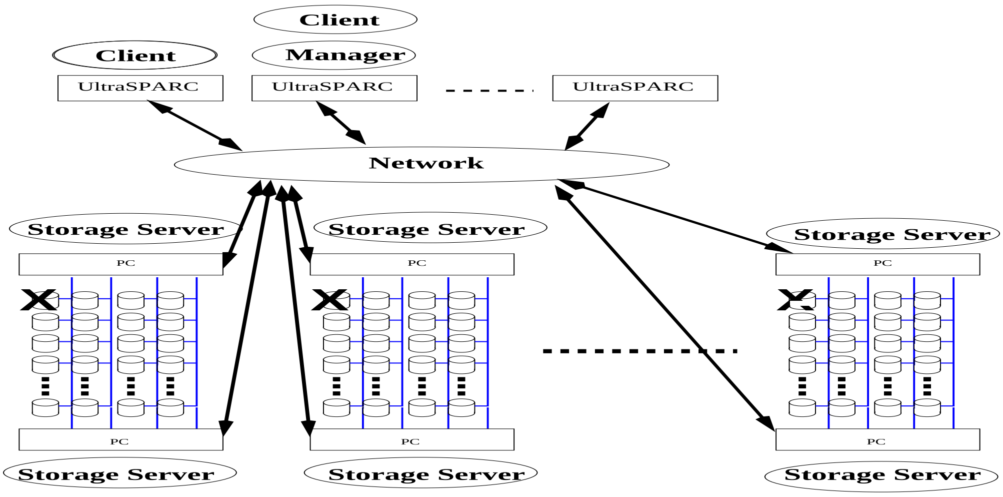

Figure 3: Organization of the PC and UltraSPARC cluster. The disks labeled with an 'X' belong to the same stripe group.

tional independent components introduced by the Tertiary Disk design do not lead to a less reliable system.

When a disk fails, all data on it is lost. When a PC fails, all disks are accessible through its double ended pair. The same is true when a SCSI controller fails. When a SCSI string fails, all disks on that string are inaccessible. In the configuration in figure 3, all nodes contribute 1 disk to a parity group. The size of all parity groups is 10. Note that as the two node designs have different numbers of disks, it is not possible to have all the parity groups be the same size and have only one disk from each node. However, since xFS stripe groups can be changed dynamically, we assume that at any time some number of disks will not be used or can be available as hot spares. Two disks in any parity group have to be inaccessible for data loss. This cannot happen through the failure of any single component in the system. The possible two component failures which can cause it are two strings or two disks in the same parity group. We do not consider any failures of three components as the probability of this is insignificant. Note that in this organization, it is possible for two PCs or two SCSI controllers to fail without any data becoming inaccessible.

Even though disk manufacturers quote Mean Times to Failure (MTTF) on the order of a million hours, in practice it is closer to 250,000 hours. The MTTF of SCSI strings (cables) is around 21,000,000 [11]. Since we assume the existence of hot spares, the Mean Time to Repair (MTTR) of a disk is only the time for reconstruction. We assume MTTR for both the disks and the SCSI strings to be 24 hours. (Note that this assmuption is very generous as, with hot spares, a disk's contents can be reconstructed in a fewhours). Using the techniques of [11], the Mean Time to Data Loss of the system becomes 723,336 hours, or approximately three times that of a single disk.

This analysis shows that, if parity groups are defined to contain only one disk per node, the resulting system is very reliable. In particular, data is accessible through PC failures, which are likely to be more common than disk or cable failures. This also shows that striping and parity features built into a small disk array may become useless if many such arrays are connected in this type of environment. If the host to the array fails, data redundancy within it will not be of any use. Other features of custom built arrays which make them attractive are hot pluggability of disks and failure monitoring. Hot pluggability of disks is supported by commodity disk enclosures. Monitoring services provided by a single disk array are usually in the form of a status display. This is less useful in a cluster of disk arrays as more centralized monitoring would be ideal. To do this it would be necessary for the disk array to provide its failure information to its host. While individual disk arrays have nice characteristics to improve reliability, they become less useful in a cluster of such arrays.

5. Cost Analysis

In this section we present a cost analysis of the architecture in section 3. The cost analysis will show that Tertiary Disk systems can be built for a small extra cost over the disk cost. Although we present only the cost of our prototype, the important point is that the cost of the infrastructure necessary to create a large storage system is a fraction of the cost of the actual storage. The cost breakdowns are based on the prices for each component as of late 1996.

Table 1 shows the cost breakdown for the two node designs. The complete system has eight nodes of design 1, two nodes of design 2, and additional infrastructure including network switches, racks, and a UPS. Each node design has 2 PCs, 2 network interface cards, and a variable number of disks, disk enclosures, cables and SCSI controllers. For node design 1, the disks account for 72% of the total node cost. The disk enclosures are 11% of the total cost, and the remaining infrastructure is 17%. In node design 2, disks form 76% of the total cost, while the disk enclosures are 14% and the remaining infrastructure is 10%. In both nodes, the disks make up the largest part of the cost, followed by the disk enclosures. As the disks/PC ratio changed from 16 in design 1 to 28 in design 2, the relative disk cost only changed from 72% to 76%. The relative costs of the disk enclosures also went up, while the relative cost of the rest of the infrastructure went down. The reason is that, when the disk/PC ratio changes in the design, the changes in the actual components is mostly in the disks and the disk enclosures. The PC and network interface cost, which is the largest cost of the remaining infrastructure, remains fixed. In comparison, the cost of additional SCSI adapters and cables is small. Put another way, node design 2 makes more use of the two PCs in the sense that all SCSI strings are fully utilized (14 disks per string). In this sense, the PCs in node design 1 are underutilized as their SCSI strings are only partly populated (8 disks per string). The performance trade-offs of the two designs will be covered in section 6.

		Node Design 1		Node Design 2	
Component	**Cost Each**	**Number**	**Total**	**Number**	**Total**
PC	**$3,000**	**2**	**$6,000**	**2**	**$6,000**
Myrinet Interface	**$1,260**	**2**	**$2,520**	**2**	**$2,520**
SCSI Controllers	**$395**	**4**	**$1,580**	**4**	**$1,580**
Cables	**$79**	**8**	**$632**	**16**	**$1,264**
Disk Enclosures	**$1,866**	**4**	**$7,464**	**8**	**$14,928**
Disks	**$1,487**	**32**	**$47,584**	**56**	**$83,272**
Totals			**$65,780**		**$109,564**
		Complete System			
		Number		**Total**	
Node Design 1	**$65,780**	**8**		**$526,240**	
Node Design 2	**$109,564**	**2**		**$219,128**	
Uninteruptible Power Supplies	**$13,380**	**1**		**$13,386**	
Network Switches	**$2,100**	**12**		**$25,200**	
Racks\Shelves	**$350**	**10**		**$3,500**	
Totals				**$787,454**	

Table 1: Costs of components.

The complete system has the additional costs of network switches, racks, shelves, and a UPS. With these additions, the disk costs form 70% of the overall system cost. The enclosures cost 11%, and the network infrastructure (switches and interface cards) form a large amount of the remainder. This analysis shows that different configurations of this architecture can be built for a small extra cost over the raw disks.

Assuming that the capacity of the system (i.e. the number of disks) is held constant, it is possible to increase or reduce the number of PCs for cost/performance. At the very extremes, very large or very low disk/PC ratios will trade off performance for cost. However, it is important to note that the number of disk enclosures, and number of SCSI controllers cannot be changed dramatically. Therefore the cost benefits from very high disk/PC ratios is not worth a lot given the significant performance losses. The network infrastructure, on the other hand, can be changed dramatically. Our design uses a large number of switches for reliability and to provide many high bandwidth links out of the system. It is possible to connect the same number of PCs with fewer switches for lower performance at lower cost.

We have not included the cost of maintenance in this analysis. Studies suggest that the cost of maintaining storage is comparable, if not larger, than the cost of the storage itself. [12] However, this maintenance cost is hard to estimate, and will exist for any comparable capacity system. Section 4 showed that monitoring and maintenance issues for a cluster of disk arrays will be similar to that of Tertiary Disk. The same argument applies for the additional cost in software complexity, which we have also not taken into account.

6. Performance of nodes

In this section we present some preliminary performance measurements of the two node designs. As of the time of this writing, the components of xFS are not ready for benchmarking. As the storage server uses the raw disk interface to access the disks, we present here some measurements of raw disk performance of the nodes. The workloads we consider are random reads of 8KB, 64KB and 256 KB. The measurements are sufficient to show the capabilities of the hardware and the performance tradeoffs between configurations.

Figure 4 shows the throughput of various sized requests on a single ended PC with 4 SCSI controllers. As Solaris does not support tagged queuing, multiple outstanding requests to a single disk do not increase throughput. Therefore the tests used only one process per disk in the configuration. On the X axis the number of disks is varied as a multiple of 4. For the 8KB requests, the bandwidth scales with the number of disks up to 32 disks. In this range, each disk achieves about 85 IO/s. This limit comes from the seek and rotational latencies. After 32 disks, the bandwidth levels off at 20MB/s. For the larger request sizes, the throughput increases to about 65MB/s and levels off. At 65MB/s, each string is delivering approximately 16MB/s. In comparison, the peak bandwidth we have observed on a single string is about 17MB/s. These measurements show that a single PC is capable of supporting the full bandwidth of 4 Fast-Wide SCSI strings. Similar experiments with disks on 5 SCSI strings have shown possible peak bandwidths of 80MB/s.

Figure 5 shows the performance of double ending. 8KB, 64KB, and 256 KB/s requests are simultaneously issued by the two PCs on a single node. The X axis gives the number of *disks accessed by a single PC*. The number of disks is varied as a multiple of 4. For each request size there are three lines in the graph, the throughput of a single PC (marked as *single*) and the combined throughput of both (marked as *double*). The vertical lines show where the two node designs fall on the graphs. At 16 disks/PC (Node 1), both the 64KB and 256KB requests have reached their peak of 35MB/s per PC. At 28 disks/PC (Node2), the bandwidth for the 64KB and 256KB requests remains at about 35MB/s. Figure 4 already showed that the bandwidth of 64KB and 256KB requests are limited by the SCSI strings. Since these strings are now shared, each PC gets approximately half of the total bandwidth. For 8KB requests, the bandwidth at 16 disks/PC (Node 1) is 11MB/s or about 85 IO/s/disk (the peak per disk). At 28 disks/PC the bandwidth is 17MB/s or 73 IO/s/disk. The 8KB request performance is not SCSI limited, and is less affected by double ending.

As the two graphs show, for the 8KB request case, a single PC accessing 28 disks got 18MB/s on unshared SCSI strings and 17MB/s when sharing SCSI strings . However, for the larger request sizes, each PC achieves only about half of the previous bandwidth. While the number of disks being accessed is small enough or the request size is small enough, double ending can increase performance. When either is

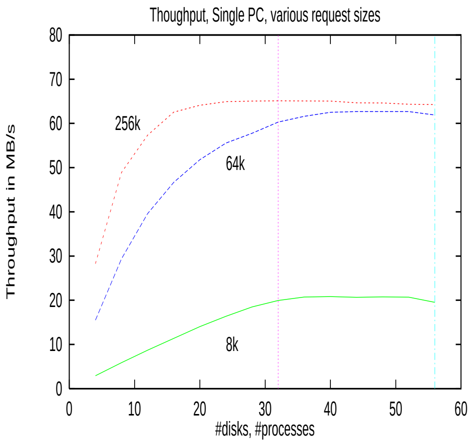

Figure 4: Throughput for 8KB, 64KB and 256 KB reads on single ended PC.

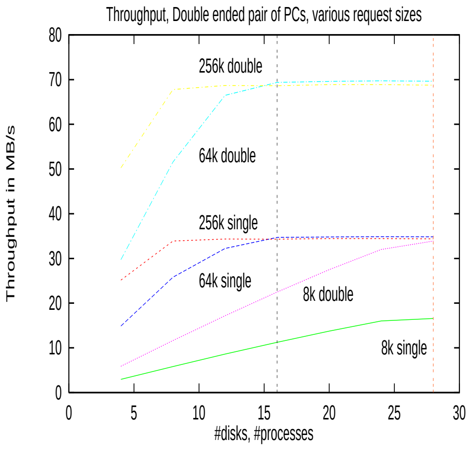

Figure 5: Throughput for 8KB, 64KB and 256KB reads on a double ended pair of PCs.

large enough that the SCSI strings become the bottleneck, sharing them causes each PC to only get about half of the possible bandwidth. Other performance issues on double ending are given in [5].

Figure 4 also shows the possible performance in failover mode (if a PC has failed and all disks in a node are accessed by one machine). The two vertical lines in the graph show where node designs 1 and 2 fall in this case. For large request sizes, a single PC is able to deliver almost as much bandwidth in failover mode as both PCs in normal mode. This is true for both node designs. For small requests, a single PC in a node of design 1 can scale upto 32 disks. A single PC in a node of design 2 cannot give the same performance for small requests in failover mode as two PCs in normal mode.

7. Discussion

The performance measurements can be used to understand some trade-offs in changing configurations. For instance, a single PC can scale up to 32 disks for a workload of 8KB requests. For larger request sizes, the PC eventually becomes a bottleneck at 65MB/s, and the number of disks sufficient to reach this bottleneck varies with the request size. For the 4 SCSI controller case, the peak was achieved at 16 disks for 256K requests and at 44 disks for 64K requests. If the requests are large enough, a single disk can deliver about 10MB/s. At this point, a PC can only deliver the bandwidth of 6-7 disks. Node designs 1 and 2 differed in relative disk cost by only 4%, but in much larger amounts in performance for large requests. Varying the disk/PC ratio is going to affect performance less for small sized requests. For larger sized requests, decreasing the disk/PC ratio makes more of the underlying disk bandwidth available, at an increased cost.

8. Related Work

The RAID project showed that large numbers of small disks can be used to build a larger system where striping is used to increase bandwidth and redundancy to improve reliability. Several projects have extended the ideas in the RAID work, which was based on a centralized controller, to networked storage systems. Two examples are TickerTAIP [13] and Petal [14]. TickerTAIP developed a fully distributed storage system with striping and RAID level 5 redundancy. Their goal was to distribute the functionality of a single centralized RAID controller across multiple nodes. There are *worker nodes,* which manage the disks, and *originator nodes,* which communicate with the client. One of the main differences between this and xFS is that the work of calculating parity is distributed among the worker nodes. This leads to more complicated failure recovery, as the worker nodes need to cooperate on all write operations. The complexity goes up with the number of nodes and number of concurrent writes in the system. In xFS the parity computation is done at the client, which simplifies the design of the storage server and leads to less communication between storage servers. The result is a more scalable system. Petal developed a distributed mirrored system. Fault tolerance is simpler in this case than for RAID 5, and the resulting system is more scalable. It also makes storage management easier by providing automatic redistribution of data when a new component is added. The main limitation of Petal is the lack of a distributed file system, which limits its scalability both in number of nodes and number of disks per node. The Zebra file system also implemented distribution of parity groups accross machines [16, 17]. The main difference between Zebra and xFS is that Zebra has a centralized manager, which makes it less scalable than xFS. These studies did not look at using commodity hardware.

9. Conclusion

Our prototype shows that large reliable, high performance storage systems can be built from commodity components. Current trends in storage devices show that such systems are feasible. The needed reliability is provided by replicating the infrastructure connected to each disk. We have shown a design for a networked storage node that has better fault tolerance characteristics than a disk array. Our initial performance measurements show that PCs are a good building block for such systems. The cost analysis shows that PCs with switched networks provide a high performance infrastructure which is a fraction of the disk cost. Even though the cost of maintenance is not included, this is harder to estimate for both Tertiary Disk and an equivalent group of independent disk arrays. As shown by the Petal work, networked storage systems can be designed for easy addition of new components, making them easier to manage. The architecture makes upgrading components and expansion easier than traditional disk arrays where the disks can be purchased only from the vendor of the array. Also, use of a distributed file system allows the architecture to scale indefinitely. Previous work on networked storage systems have been limited by the lack of a truly distributed file system.

Remarkably, the argument for Tertiary Disk versus large custom built disk arrays is nearly identical to that made for clusters or Networks of Workstations (NOW) versus MPPs

[15] The NOW argument is based on just-in-time assembly (to reduce lag time for the rapidly improving microprocessors), the high cost of low volume manufacturing of MPPs, and the emergence of switched LANs. Custom designed hardware RAIDs have the same high cost and lag time. A major tenet of both studies is improving *cost* as well as performance.

Our immediate future work is to finish the integration of xFS and Tertiary Disk. Other future work includes development of a centralized monitoring facility for the full prototype, and studying data layout and backup issues for networked storage systems.

10. Acknowledgments

We wish to thank Jim Gray of Microsoft for the initial suggestion to investigate commodity storage systems. We would like to thank IBM for the donation of the disk drives used in Tertiary Disk, and Intel for the donation of Pentium Pros. This research is performed in conjunction with the NOW project and is funded by the ARPA Roboline-Storage contract.

11. References

[1]Gibson, G. Nagle D. Amiri K. A Case for Network Attached Secure Disks. Technical Report CMU-CS-96-142 Carnegie Mellon University School of Computer Science, June 1996.

[2]Hanlon W. Fener, E, Data Storage and Management Requirements for the Multimedia Computer Based Patient Record. *IEEE Mass Storage Symposium* 1995, p. 11-16

[3]Kobler, B. Berbert, J. Architecture and Design of Storage and Data Management for the NASA Earth Observing System Data and Information System (EOSDIS) I*EEE Mass Storage Symposium*, 1995 p. 65-76

[4]Shiers, J.D. Data Management at CERN: Current Status and Future Trends *IEEE Mass Storage Symposium* 1995 p.174-81

[5]Talagala, N. Asami S. Patterson D. Double Ending Implementation. University of California at Berkeley, Computer Science Division.

[6]R. Wang and T. Anderson. xFS: A Wide Area Mass Storage File System. In *Fourth Workshop on Workstation Operating Systems*, pages 71–78, October 1993.

[7]Anderson, T. Dahlin, M. Neefe, J. Patterson, D. Roselli, D. Wang R. Severless Network File Systems. *15th Symposium on Operating Systems Principles* pp 71-8. De-

[8]Chen. P.M. Lee, E.K., Gibson, G.A, Katz, R.H. Patterson, D.A. RAID: High Performance Reliable Secondary Storage. *ACM Computing Surveys* June 1994 vol.26 {no.2):145-88

[9]Chen, P. Gibson, G. Katz, R.H Patterson, D.A. Schulze, M. Two Papers on RAIDs, Report UCB/CSD 88/479 Computer Science Division University of California at Berkeley

[10]Schulze, M.E Considerations in the Design of a RAID Prototype Technical Report UCB/CSD 88/448, University of California at Berkeley, 1988

[11]Gibson, G. Redundant Disk Arrays, *Reliable Parallel Secondary Storage*. PhD Thesis, University of California at Berkeley, 1990.

[12]Lee, E. Petal, Highly Available, Scalable Network Storage. *Proceedings of COMPCON* 1995.

[13]P. Cao, S. Lim, S. Venkataraman, and J. Wilkes. The TickerTAIP Parallel RAID Architecture. In *Proc. of the 20th Symp. on Computer Architecture*, pages 52–63, May 1993

[14]Lee, E. Petal, Distributed Virtual Disks, *Proceedings of ASPLOS* 1996.

[15]Anderson, T. Culler, D. Patterson, D. and the NOW Team. A Case for Networks of Workstations. I*EEE Micro* p54-64 February 1995

[16]J. Hartman. *The Zebra Striped Network File System*. PhD thesis, University of California at Berkeley, 1994.

[17]J. Hartman and J. Ousterhout. The Zebra Striped Network File System. *ACM Trans. on Computer Systems*, August 1995.