SANSIM: A PLATFORM FOR SIMULATION AND DESIGN OF A STORAGE AREA NETWORK

Yao-Long Zhu, Chao-Yang Wang, Wei-Ya Xi, and Feng Zhou
Data Storage Institute
DSI building, 5 Engineering Drive 1, (off Kent Ridge Crescent, NUS)
Singapore 117608
Tel: +65-6874-6436
e-mail: zhu_yaolong@dsi.a-star.edu.sg

Abstract

Modeling and simulation are flexible and effective tools to design and evaluate the performance of Storage Area Network (SAN). Fibre Channel (FC) is presently the dominant protocol used in SAN. In this paper, we present a new simulation - SANSim, developed for modeling and analyzing FC storage network. SANSim includes four main modules: an I/O workload module, a host module, a storage network module, and a storage system module. SANSim has been validated by comparing the simulation results with the actual I/O performance of a FC RAM disk connected to a FC network. The simulated results match the experimental readings within 3%. As an example of applicability, SANSim has been used to study the impact of link failures on the performance of a FC network with a core/edge topology.

1. Introduction

SAN architecture has been proven to provide significant performance advantages, larger scalability, and higher availability over traditional Direct Attached Storage architecture. It is therefore not surprising that the performance modeling and simulation of SANs has become an interesting field of research [1][2]. Xavier [3][4] used the CSIM language to simulate a SAN and model its activities. Petra et al. [5] presented the SIMLab as a simulation environment based on a network of active routers. Wilkes [6] used the Pantheon storage-system simulator to model the performance of parallel disk arrays and parallel computers. DiskSim [7] is another disk storage system simulator to support research in storage subsystem algorithm and architecture.

However, the simulation studies and tools mentioned above have been very limited in modeling and simulation at the FC protocol level. Nevertheless, it is necessary to simulate at the frame-level FC in order to monitor and analyze details of FC SAN activities.

In this paper, we present a new FC SAN simulation, SANSim, which supports FC frame-level and fully simulates Fibre Channel protocols in accordance to the relevant standards [10-13] to guarantee the compatibility and interoperability of different modules. In the following sections, we first introduce our simulation tool SANSim in section 2. Then, we present in section 3, the experimental results and simulation validation of a FC network. Finally, we simulate and analyze some FC network design issues in section 4.

2. SANSim

SANSim is an event-driven simulation tool for SAN that includes four main modules: an I/O workload module, a host module, a storage network module, and a storage system module, as shown in Figure 1.

The I/O workload module generates I/O request streams according to the workload distribution characteristics and sends them to the host modules. The host module encapsulates the I/O workload to the SCSI commands and sends them to the Host Bus Adaptor (HBA) sub-modules. The storage network module simulates the network connectivity, topology and communication mechanism. The FC network module includes three sub-modules: a FC_controller module, a FC_switch module and a FC communication module. The storage module maps I/O data to the storage devices.

SANSim is developed in pure standard C. It has been compiled successfully both in Microsoft Window XP and Linux platforms. The simulator reads in configuration parameters from a user specified input file, plots measurement data, and writes data in an output file after the simulation is completed. The simulation duration and the warm-up period can also be specified in the input file to control and eliminate the transient bias in the simulation results. The configuration parameters for each of the four modules are

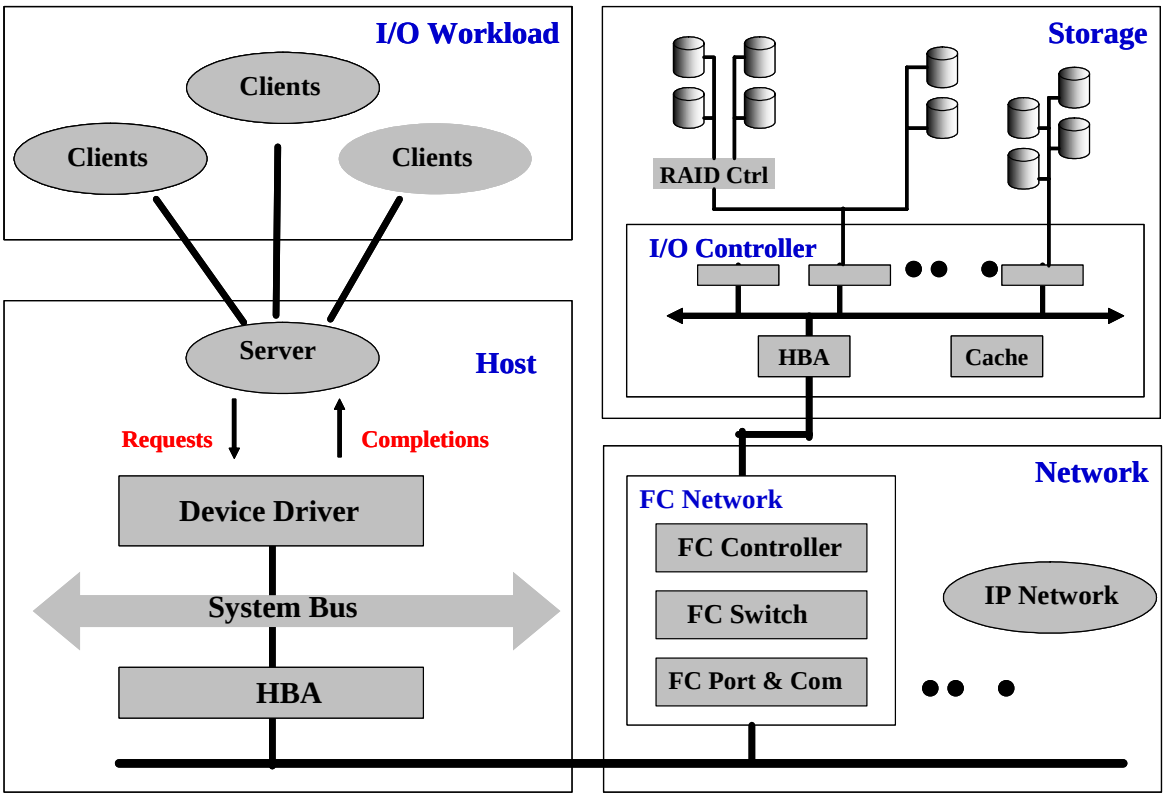


Figure 1 SANSim internal structure including I/O workload module, host module, storage network module and storage module.

arranged orderly in the input file.

2.1 I/O Workload module

The key function of I/O Workload module is to generate I/O request streams according to the workload distribution characteristics and send them to the host modules. The current module supports both system-traced I/O workloads and synthetic I/O workloads. We use a synthetic I/O workload in this paper.

Generally speaking, a disk request is defined by five dimensions: the requested pattern, the size distribution, the repeatability, the location distribution and the I/O operations. The workload module is able to generate several basic different arrival patterns such as Poisson arrivals, equal time intervals, Normal distribution arrivals and so on. It can also simulate a combination of request patterns that consists of the different distributions and different rates. Another capability of the Workload module is to generate repeatable requests. This scenario is used to define a workload in which some files are more popular than others and consequently accessed more frequently.

2.2 Host module

Host module includes a device driver, a SCSI layer, a system bus, as well as DMA modules. The main function of the Host module is to encapsulate the I/O workload into the SCSI commands and sends them to the Host Bus Adaptor (HBA) sub-modules.

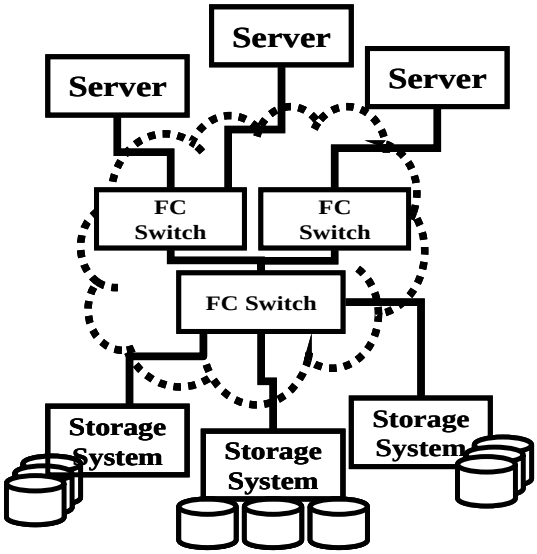
The host module schedules the I/O requests generated by the workload module, based on various schedule policies, which are configurable. The I/O requests are traced in a waiting queue and an outstanding queue. An I/O request in the outstanding queue refers to a request, which has been submitted to the storage device, but has not been completed yet. The maximum number of outstanding requests depends on the buffer size and system configuration. The I/O requests in the waiting queue may be merged or re-scheduled following certain policies.

The host module supports a multiple-host configuration. Each host has a separate I/O generation module. There is a specific mechanism to identify the I/O requests coming from different hosts.

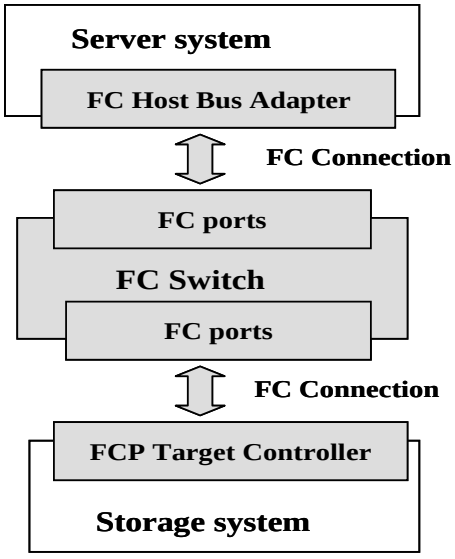
2.3 FC network module

The key function of the FC network module is to simulate the FC connectivity, topology and communication protocol. The FC network module includes three sub-modules: the FC Controller module, the FC Switch module and the FC Port & Communication module, as shown in Figure 2. The FC Controller module simulates the communication behavior of FC Commands or Data Frames. The FC Switch module models all the FC Ports, switch architecture, and as well as the routing and flow control. The FC Port & Communication module transfers FC Frames between the FC Ports.

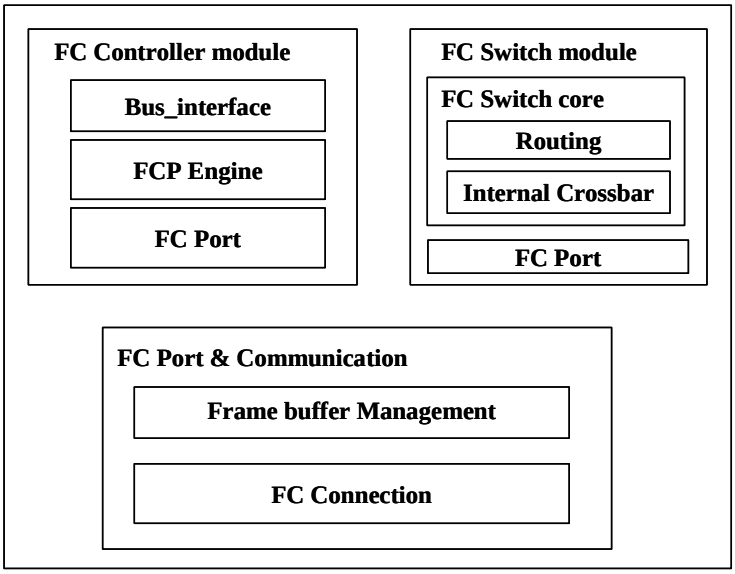
2.3.1 FC Controller module



(a) An example of FC SAN



(b) Abstracted view



(c) FC network module

Figure 2 Modeling of Fibre Channel network in SANSim

The FC Controller module models both initiator and target modes of the FC HBA. As shown in Figure 3, the FC Controller module includes three sub-modules: a Bus_interface, an FCP(SCSI over Firbre Channel Protocol [13]) Engine and a FC Port. The Bus_Interface sub-module handles the communication between the device driver and the controller such as DMA and interruptions. The FCP Engine has the responsibility of

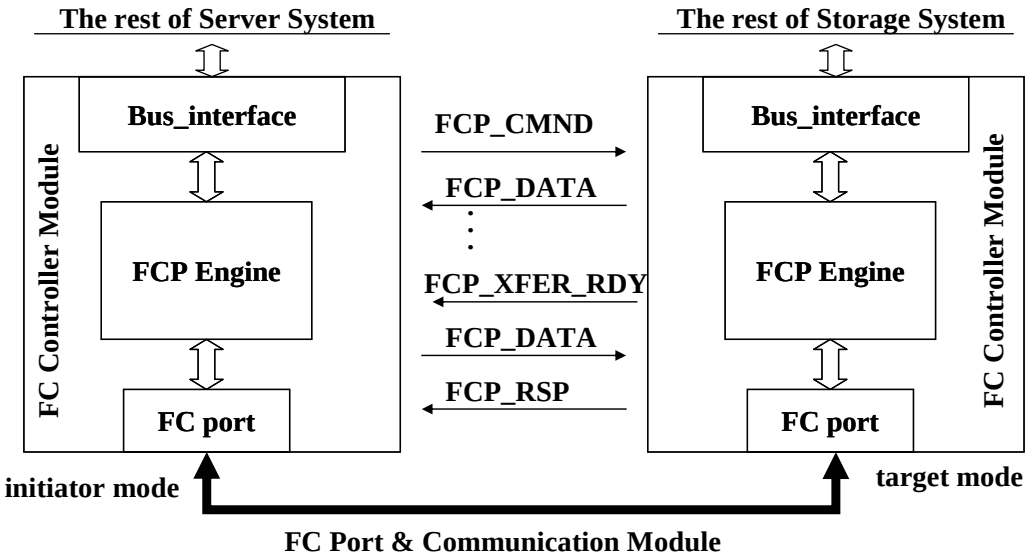


Figure 3 FCP Operation

constructing different FC frames corresponding to each sequence of FCP exchange. The FC_Port is responsible for delivering FC frames to the destination port.

When a SCSI request arrives in the initiator, the device driver sends SCSI commands to the FC_Controller through the Bus_Interface. The FC_Controller then executes the commands and fetches SCSI I/O's information from memory. A FCP_CMND frame is constructed for each SCSI I/O in the FCP Engine, and then sent out through FC_Port module. After the target FC_Controller receives the FCP_CMND, the target firmware decapsulates the FCP_CMND and processes the SCSI command.

In the case of a FCP Read operation (SCSI Read 10), the target host decapsulates the SCSI request and prepares the requested data. Once the data is ready, the target FC_Controller takes the responsibility of transferring data back to the requestor (the initiator) and sends a completion message FCP_RSP after all data is sent. With a FCP Write operation (SCSI Write 10), the target driver allocates sufficient memory area for the incoming data, and sends a FCP_XFER_RDY to the initiator requesting the data. After the initiator receives the FCP_XFER_RDY, it starts to send a FCP_DATA. Finally when the target receives all data successfully, it sends a FCP_RSP indicating the completion of the FCP operation.

2.3.2 FC Switch module

SANSim FC Switch module has two sub modules: FC port, and FC Switch core. FC port supports F_Ports/FL_Ports and E_Ports. F_Ports/FL_Ports are for host-switch and device-switch connections, and E_Ports are for switch-switch inter-connections. The FC_Port's address_ID is unique and confined to the FC-SW-2 standard [11]. FC Switch core is the switch's control center for frames routing and forwarding. It contains routing and internal cross-bar. If the destination port of requested FC frames is busy, the incom-

ing frames are held in the incoming buffer until they are successfully routed. SANSim uses Dijkstra's algorithm [14] to compute the shortest routing path. The routing table remains constant unless the network connectivity is changed during the simulation. When the network configuration is changed, the switch module re-computes the shortest path.

2.3.3 FC Port & Communication module

The FC Port & Communication module includes a Frame Buffer management and FC connection sub modules. The Frame Buffer management sub module handles all management of incoming and outgoing frame buffers. The FC connection sub module establishes a FC connection between two FC_Ports used to transfer frames.

The Fibre Channel Arbitrated Loop (FC-AL) is a typical sub FC connection sub module used in the simulation. The FC-AL connection module consists of the following sub modules: Loop Port State Machine (LPSM), Alternative BB Credit Flow Control (Alt BB Mgr), and Loop Port Process Control (LPPC). The LPSM models the L_Port's state transition during the communication. The L_Port transmits FC Frame only when it is in certain states. The number of frames permitted to be transmitted depends on the available buffer size in the destination port. The Alternative BB Credit Flow Control sub module models the flow control method defined in the Standard [10] to avoid overflow. The Loop Port Process Control sub module models the remaining behavior of the L_Port, such as responding to a certain loop port request, re-transmitting an ARBx signal and other port activities.

2.4 Storage module

The main function of the storage module is to map I/O data to the storage devices. Storage modules, which include interface module (HBA), storage controller module and storage device module, can simulate a RAID system with various cache management algorithms, disk drives, and RAM disks. In the event of disk failure in a RAID system, the degraded mode and rebuild behavior can also be modeled. Various RAID algorithms can be integrated into the storage modules.

3. Simulation Validation

3.1 Experimental environment

The experiments were conducted on an in-house developed FC RAM disk, which maps all storage I/Os to the memory rather than using an actual magnetic disk. Since we are focusing on the FC simulation and validation, using RAM disk as a target helps to isolate problems caused by the modeling of a hard disk drive. The initiator and target use a FC-AL connection. Table 1 lists the detailed hardware and software configurations used in the experiments. I/OMeter, a widely accepted industry standard benchmark tool, is used here to collect experimental data. The monitored parameters include IOPS (I/O per second), throughput (MB/s), queue depth, I/O request size, and read/write operations. The IOPS is used primarily with small I/O requests, while the throughput is used with large I/O requests. The queue depth refers to the number of outstanding I/O requests, which are injected into the FC network and storage system. The I/OMeter issues a number of re-

Table 1 System configuration for SANSim FC-AL module validation

Initiator	
Hardware	CPU: AMD AthonMP 1600+ FC HBA: Qlogic 2300 RAM: 2x256MB DDR SDRAM Main board: 64 bit PCI Tyan Tiger MP2466N
Software	OS: Windows XP Professional SP1 Driver: Qlogic Driver Version 8.1.5.12 Tool: Intel IOMeter Version 2003.02.15
Target	
Hardware	CPU: Intel PIII 1GHz FC HBA: Qlogic 2300 RAM: 4 x 1GB Kingston ECC Reg. PC133 Main board: 64bit PCI, Supermicro 370
Software	OS: RedHat 8.0 Kernel: 2.4.18 Driver: In-house 2300 target driver Version 1.0, In-house Linux RAM Disk Version 2.0

quests (equals to the queue depth) initially, and generates new I/O requests only after the completion of previous requests. Fixed I/O sizes were used in all requests.

3.2 Comparisons of the experimental and simulated data

Figure 4 and 5 show that I/O transaction performance varies with the queue depth with read and write operations. The I/O sizes are set to 2KB, 8KB, 16KB, and 32KB respectively. The IOPS increases with the queue depth and then reaches a saturation limit. In other words, there is a critical queue depth. When the queue depth is bigger than this critical value, the system transaction performance shows no improvement and the response time for I/O request becomes worse. For example, when an I/O size is 8KB and IOMeter sends 100% of read operations, the maximum transaction performance is 17.5k IOPS. This translates into a network and storage overhead means of around 0.057ms (1second/17.5k). Simulation results show that the average response time is about 0.471ms when queue depth equals to eight. The largest contribution in the response time is from the queue waiting time, not the serving time.

Figure 6 and 7 show that the throughput (MB/s) varies with the I/O request size with read/write operations. The queue depths are set to 1, 2, and 8 respectively. Generally, the throughput increases with I/O request sizes. When the request size is large enough (more than 128KB with queue depth of 8), the data transfer time dominates the overall overhead. Then the throughput is limited by the FC network bandwidth.

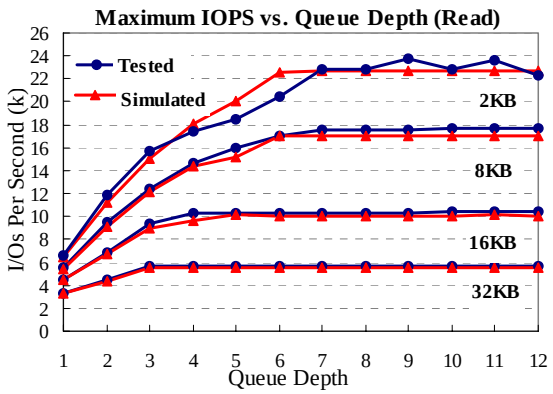


Figure 4 Read performance vs. queue depth

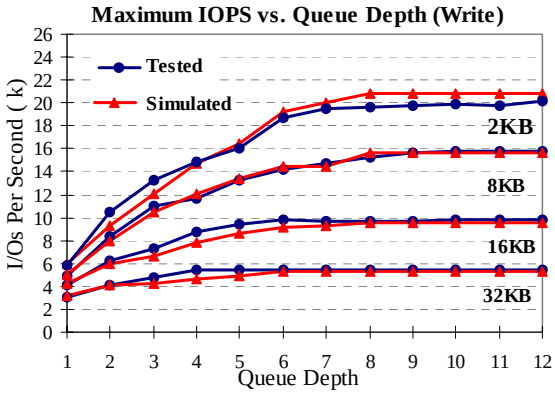


Figure 5 Write performance vs. queue depth

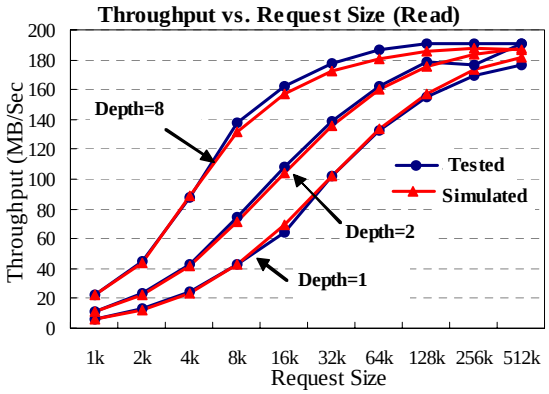


Figure 6 Read throughput vs. I/O size

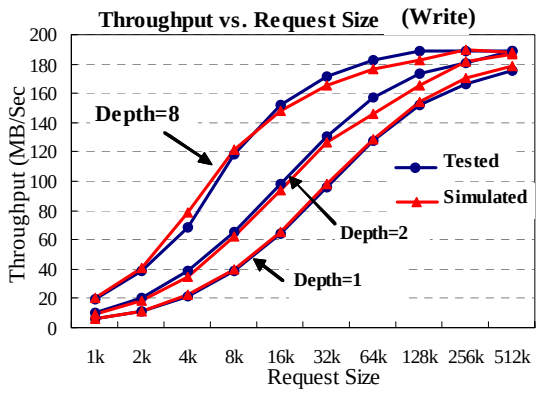


Figure 7 Write throughput vs. I/O size

The simulation results and experimental data match very well in all cases as illustrated in figures 4-7. The error range is generally less than 10%. With read operations, it is less than 3%.

4. FC Network simulation and analysis

4.1 Simulation Environment

To illustrate an application of SANsim, the performance and availability of a FC network are simulated and analyzed in this section. We conducted a simulation based on the core/edge network with five FC switches, as shown in Figure 8. FC switches 1-4, as edge switches, are connected to the core switch 5 through two 2G FC ISLs. Each FC controller has 32 frame buffers. Each FC port on a switch has four frame buffers. The distance between any two points in the network is set to be 50 meters. The storage controller processing capacity is 22.5K transactions per second.

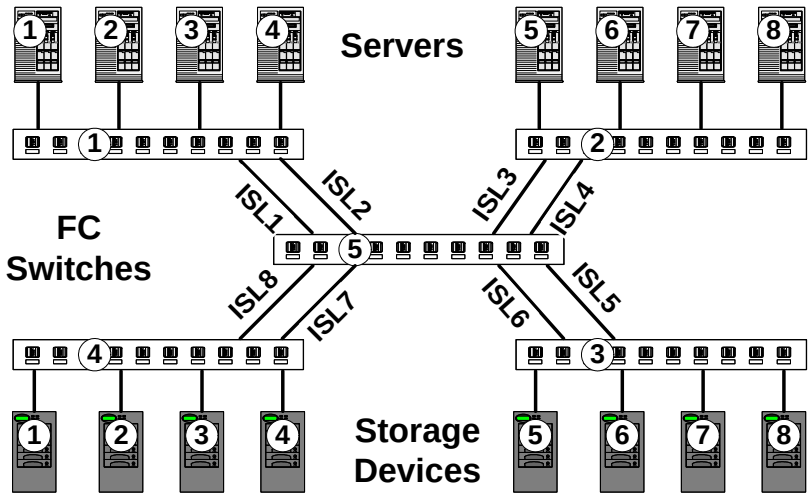


Figure 8 Simulation configuration

A synthetic I/O workload is applied to each server. The I/O inter-arrival time follows an exponential distribution. The maximum number of outstanding I/O requests is 32 for each server. Since the simulation is based on an open system, the I/O queue in the server is allowed to grow without limitation. However, the servers issue a new request to the network and storage devices only after a previous request is completed. The IO requests are randomly and evenly allocated among all devices.

4.2 Simulation Results and Analysis

In order to study the impact of link failure on the network throughput, we conducted a series of simulations under four different scenarios: no link failure, ISL1 failure, ISL8 failure and simultaneous failures of ISL1 and ISL8. The throughputs of all servers are collected using 2KB and 32 KB request sizes. Servers 1 through 4 have the same characteristics and achieve similar throughput results. We use an average throughput S1-4, as shown in Figure 9, to represent the performance of Server 1 though 4. Same process is applied to server 5 through 8.

Figure 9(a) shows the throughputs as a function of I/O workload for case I (without link failure). The throughputs grow linearly and then reach asymptotic values. All servers achieve the same throughputs due to the symmetrical characteristics of the network. When the I/O size is 2KB, the maximum throughput is 45MB/s. Since the process capacity of the storage device is 22.5k transactions per second, it limits the maximum throughput to 45MB/s for 2KB I/O size. When the I/O size is 32KB, the maximum throughput for each server is 80MB/s. The total throughput for a single network link is 160MB/s and that is 20% less than the nominal value 200MB/s. The simulated results show that the maximum throughout supported by a single storage device is 175MB/s for 32KB I/O size. That means the performance is not limited by the storage devices, but by the network.

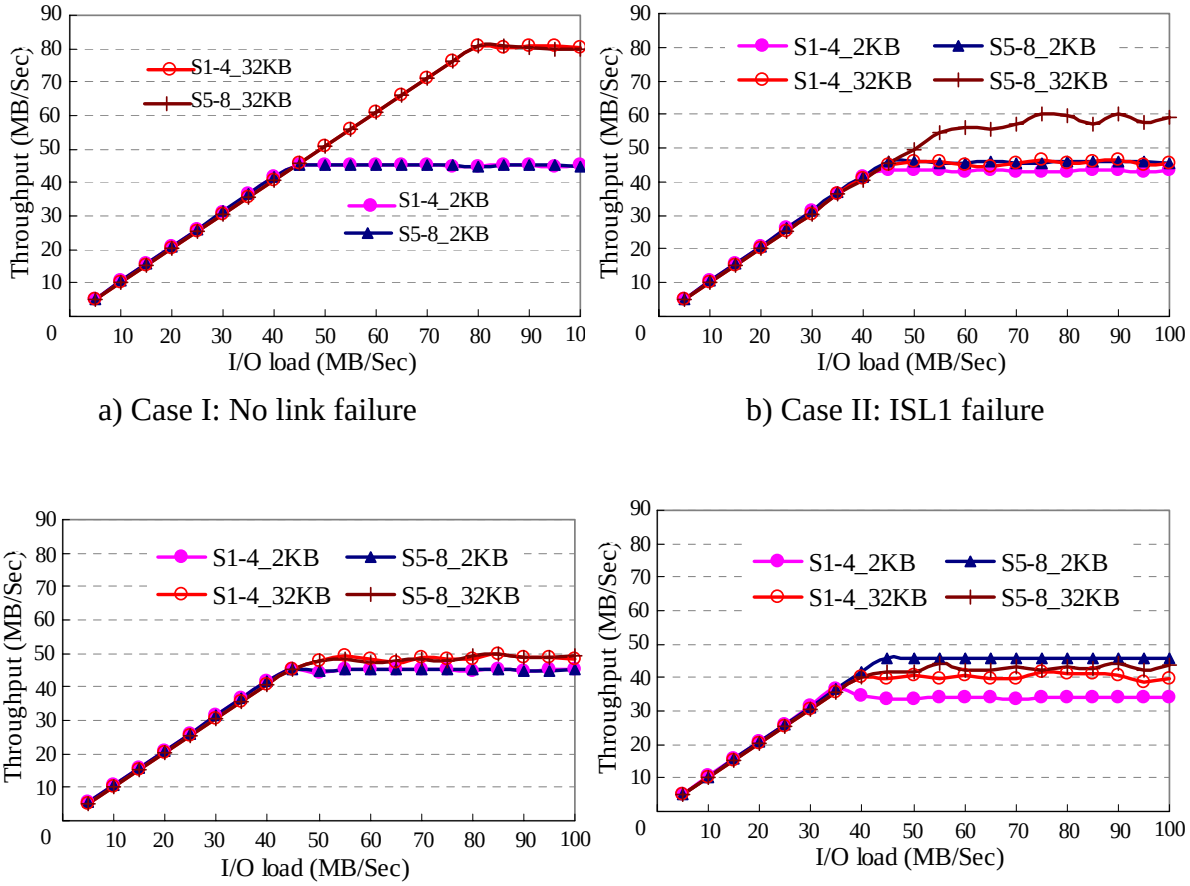


Figure 9 Maximum throughputs under symmetrical I/O load for different cases

Figure 9(b) shows the throughput for the case II (with ISL1 failure). When the I/O size is 32KB, the asymptotic throughput of servers 1~4 drops to 46MB/s, compared to 80MB/s in case I. Apparently, this is due to the limited bandwidth of a single link ISL2. It is also noted that a large performance drop happens with servers 5~8 (58MB/s vs. 80MB/s) compared to case I, even though the ISL1 has no direct physical connection to those servers. This decrease in performance is probably caused by the head-of-line blocking [15]. The data traffics from storage devices to servers 5~8 are competing at the core switch 5 with the data traffics from storage devices to servers 1~4. When the I/O size is 2KB, the throughput of servers 1~4 approaches to 44MB/s which is slightly less than the 45MB/s in the case I. This is caused by the competition of data traffic on the highly utilized ISL2 link. The bandwidth utilization of ISL2 reaches 176MB/s over 200MB/s.

For case III (with ISL8 failure), the maximum throughputs for servers 1~4 and servers 5~8 drop from 80MBs in case I to an average of around 50MB/s, as shown in Figure 9(c), when the I/O size is 32KB. The reason why servers 1~4 and servers 5~8 achieve the same maximum throughput is because the data traffics from storage devices 1~4 to servers 1~8

are equally affected by ISL8 failure. When the I/O size is 2KB, the maximum throughput is almost not affected by the ISL8 failure compared to case I.

When both ISL1 and ISL8 fail (case IV), the measured throughputs of servers are shown in Figure 9 (d). When I/O request size is 32KB, the asymptotic throughput of server 1-4 reaches 40 MB/s while the throughput of servers 5-8 is about 42MB/s. However, when the I/O size is 2KB, the throughput of servers 1~4 is only 35MB/s, and servers 5~8 is 46MB/s. In order to analyze the detailed frame activity on the network, data traffic across ISL1~4 and ISL5~8 are monitored and the average I/O response time are measured. The results show that the response time of the I/O issued by servers 1~4 to storage devices 1~8 becomes significant long (>2ms) when the I/O workload is larger than 36.5MB/s. This allows the storage devices to serve more I/O requests issued by servers 5~8. So servers 5~8 can achieve better performance than servers 1~4 when the I/O size is 2KB. However, when the I/O size is 32KB, the response time of I/O requests issued by servers 5~8, increases notably due to the effects of header-of-line blocking. It limits the performance of the servers 5~8 to 42MB/s with 32 KB I/O size.

5. Summary and future work

In this paper, we have presented SANSim, a platform for simulation and design of a FC SAN. The SANSim, which is based on FC frame level, can simulate all primitive signals (IDLEs, RDYs etc), commands and data frames. The design of SANSim is modular and scalable. Such tool is useful to the rapid development of high-end SAN due to the ever-increasing complexity of the SAN architecture.

We have conducted several experiments to compare experimental and simulated results. The results show that SANSim model is accurate within less than 3% in read operation, and less than 10% in write operation. As an example, the performance and availability of a core/edge FC network has been analyzed. The simulation results show that the core/edge topology suffers from certain level of bandwidth loss due to the Head-of-Line blocking caused by traffics crossing multi-stage switches. Generally, the maximum throughput achieved at all servers decreases when link failure happens. The servers on different locations have different I/O performance sensitivity to the link failure.

Future development work of SANSim includes IP storage module, Object-based storage module, file system simulation.

Reference

[1] Yao-Long Zhu, Shun-Yu Zhu and Hui Xiong, " Performance Analysis and Testing of the Storage Area Network", 19th IEEE Symposium on Mass Storage Systems and Technologies, April 2002.

[2] T. Ruwart, "Disk Subsystem Performance Evaluation: From Disk Drivers To Storage Area Networks", 18th IEEE Symposium on Mass Storage Systems and Technologies, April 2001.

[3] Xavier Molero, Federico Silla, Vicente Santonja and José Duato, "Modeling and Simulation of Storage Area Networks", Modeling, Analysis and Simulation of Computer and Telecommunication Systems, IEEE 2000.

[4] Xavier Molero, Federico Silla, Vicente Santonja and José Duato, "A Tool For The Design And Evaluation Of Fibre Channel Storage Area Networks", Proceedings of 34th Simulation Symposium, 2001.
[5] Petra Berenbrink, André Brinkmann and Christian Scheideler, "SIMLAB - A Simulation Environment for Storage Area Networks", 9th Euromicro Workshop on Parallel and Distributed Processing (PDP), 2000.
[6] John Wilkes, Richard Golding, Carl Staelin, and Tim Sullivan, “The HP AutoRAID Hierarchical Storage System”, ACM Transactions on Computer Systems, 1996.
[7] Gregory R. Ganger and Yale N. Patt., “Using System-Level Models to Evaluate I/O Subsystem Designs”, IEEE Transactions on Computers 1998.
[8] Thomas M. Ruwart, “Performance Characterization of Large and Long Fibre Channel Arbitrated Loops”, IEEE Network 1999.
[9] John R. Heath and peter J. Yakutis, "High-Speed Storage Area networks Using Fibre Channel Arbitrated Loop Interconnect", IEEE Network 2000.
[10] FC-AL, "FC Arbitrated Loop," ANSI X3.272:1996
[11] FC-PH, "Fibre Channel Physical and Signaling Interface (FC-PH)", ANSI X3.230:1994.
[12] FC-SW, “FC Switch Fabric and Switch Control Requirements”, ANSI X3.950:1998.
[13] Technical Committee T11, FC Projects http://www.t11.org/Index.html.
[14] E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs”, Numerische Mathmatik 1 (1959) 269-271
[15] M. Jurczyk, “Performance and Implementation Aspects of Higher Order Head-of-Line Blocking Switch Boxes”, Proceedings of the 1997 International Conference on Parallel Processing, IEEE 1997