Ethernet-based Network Storage – Reality or Pipe Dream?

Patrick B. T. KHOO, Wilson Y. H. WANG and H. N. YEO
MCSA Group, NST Division – Data Storage Institute
Affiliated to the National University of Singapore
Funded by the Agency for Science, Technology And Research Singapore
DSI Building, 5 Engineering Drive 1, 117608 Singapore
Email: {khoo_beng_teck, wang_yonghong, yeo_heng_ngi}@dsi.a-star.edu.sg
Web: http://nst.dsi.a-star.edu.sg/mcsa/

Abstract

The basic problem in network storage today is how to implement solutions that are cost effective and efficient but without using a lot of new components and equipment. Implementing Storage Area Networks today generally implies using Fibre Channel, however, a lot of work is being done to try to use Ethernet instead for network storage [1][2][3]. But these new Ethernet solutions still seem a little short.

In this paper, we would like to first introduce some of the key features of the new open source network storage protocol, HyperSCSI, and how it is different from existing solutions. This will be followed by benchmark and test results to show how HyperSCSI is capable of using an existing Ethernet-based network infrastructure, common-off-the-shelf hardware and well-established storage technologies and turning that into a high-performance and reliable network storage solution. Since specialized hardware and custom software are not required for HyperSCSI, we believe this is a step in the right direction to building cost effective and efficient network storage solutions.

Finally, we want to draw just one conclusion in our paper, that the existing network infrastructure and technologies can be successfully exploited to meet the requirements of network storage. The lesson learned in the search for this answer is that one must be ready to look for innovative new methods and perhaps, use unconventional thinking to meet the requirements of network storage. The resultant HyperSCSI protocol is proof of this.

1 Introduction

The concept of a network for storage has not been a new one. Before the terms “Storage Area Networks (SAN)” and “Network Attached Storage (NAS)” became commonplace, mainframes, and later on, simple file servers have been doing “network storage” for years. Of course, it was not quite in the form that we recognize today, but it was certainly a kind of network-ed storage. In fact, the very concept of transferring data over a wire, which is fundamental to network storage, is older than “networking” itself.

Fibre Channel (FC) is today a key technology for the deployment of the modern SAN. The explosive growth of the Internet, applications like CRM and ERP and so on, has been supported in part by the ability of storage to scale. However, its adoption has not been as stellar as many had hoped. There was even a time when people talked about running IP over FC, but not anymore. The deployment of SANs has not grown as fast as the analysts had predicted. Of course, part of the reason for this is due to socio-economic events and downturns, but certainly not the whole reason.

With new advancements like Fast and Gigabit Ethernet (and 10GE around the corner), as well as new high performance wire-speed Layer 3 switching and so on, it is a good time to analyze if Ethernet-based network storage can do the job. However, it is not as simple, nor as easy as that. A lot of work needs to be done and the industry developed various technologies like iSCSI, mFCP, iFCP, FCIP and iSNS to fill this gap [4]. The two leading core protocols are of course Internet SCSI (iSCSI) [5] and Fibre Channel over IP (FCIP) [6]. While they may seem to compete, they actually meet different needs. FCIP is needed for bridging FC-SANs over an IP-based network, while iSCSI is more for storage over an IP network. iSCSI is probably the leading contender that allows the deployment of pure Ethernet-SANs without the use of FC. However, its performance has not been adequate. To solve this problem, development work has shifted towards hardware acceleration [7] like TCP/IP Offload Engines (TOEs) and iSCSI HBAs. These new developments are needed to push iSCSI to comparable FC speeds. But the reality is, with all these add-ons, what will be the difference between a TOE-NIC and an iSCSI HBA from a FC HBA then? TOE and iSCSI HBA manufacturers point to the cost savings you will get by using Ethernet switches instead of FC-based switches, and the fact that TOEs and iSCSI HBAs will get cheaper as adoption increases. But does that sound a little like vapor-ware? As a result, adoption is still rather slow. Perhaps a new solution is required.

Designing a network storage protocol is not as straightforward as it might seem. We must consider that the characteristics of data storage are different from the conventional data traffic. Furthermore, the quality of the network has improved greatly compared to many years ago. As such, we believe that a non-conventional approach to solving this problem is required. In this paper, we present precisely such a solution, a new open source network storage protocol, which we have named HyperSCSI [8].

HyperSCSI is designed for the transmission of SCSI commands and data across a network in a simple and efficient way. The current implementation runs over an Ethernet network, uses existing common-off-the-shelf hardware and components and does not require any additional customized software or expensive hardware accelerators. This ultimately, will reduce the cost of an overall network storage implementation. We believe that using existing hardware and technologies does not compromise on performance and reliability. In this paper, we will present key features of HyperSCSI and how it is different from existing solutions as well as various test results to demonstrate its capabilities.

2 Selected Key Features of HyperSCSI

HyperSCSI is a new open source network storage protocol designed for the transmission of SCSI commands and data across a network. To put this in “ordinary” terms, it can allow one to connect to and use SCSI and SCSI-based devices (like IDE, USB, Fibre Channel) over a network as if it was directly attached locally. This section focuses on a few key features of the HyperSCSI protocol, and how they differ from existing solutions.

2.1 Device Discovery Mechanisms – Using the HyperSCSI Group Name

To identify and locate storage devices, Fibre Channel uses the World Wide Name (WWN) mechanism while iSCSI/FCIP/iFCP uses iSNS [9]. Such mechanisms are complex and add another hindrance to achieving ease of use and even plug-and-play networking. For this purpose, HyperSCSI uses a standard Ethernet broadcast mechanism for device discovery but adds a Group Name to segregate servers and clients. This allows clients to dynamically locate targets on the network.

If a server is configured to be in the same group with a client, it will respond appropriately, otherwise the device discovery request is ignored. Thus the only configuration users have to be concerned about is granting permissions, rather than setting up complex name servers of some type. This is particularly useful in a plug-and-play wireless personal storage network environment. This also means that there is no single point of failure like having iSNS servers or requiring expensive switches with additional intelligence built-in. HyperSCSI clients will then attempt to connect to the servers in the groups given to it, and no other.

For example, as shown in Figure 1, Server A has 2 disks and Server B has 1 disk that can be exported. Server A exports Disk 1 to “Group ABC” and Disk 2 to “Group DEF”. Server B also exports its one disk to “Group DEF”. Client X can access Disk 1 of Server A only, since it has access to “Group ABC”. However, Client Y being configured to look in “Group DEF” can see both Disk 2 on Server A as well as Disk 1 on Server B.

Groups are secured through HyperSCSI's authentication mechanisms. This example shows that HyperSCSI Group Names are flexible and easy to use.

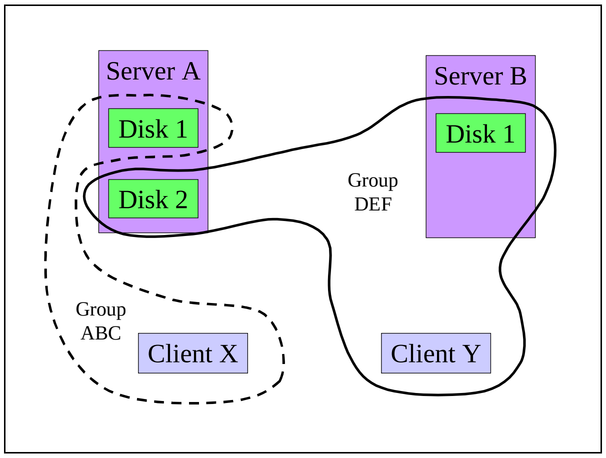

Figure 1: HyperSCSI Group Names Example

2.2 Flow Control Mechanisms

Conventional network protocols, especially TCP/IP, transfers data in streams and use an acknowledgement based sliding window mechanism for flow control and packet retransmission. This method works quite well for the uncertain ("worst case") network conditions, like telephone networks. In the SCSI world however, the host adapter initiates a connection to a device, knows precisely the capabilities of the device, and then applies this knowledge to a dedicated channel. Thus, for the SCSI protocol, a default credit-based flow control mechanism is used. HyperSCSI adopts a moving window mechanism but makes the window size dynamic. A balance is provided in that the window size does not fluctuate like TCP/IP's sliding windows, but can and does change dynamically in the middle of a connection. Since clients and servers dynamically control the window size, algorithms for determining the window size can be adopted to find the optimal window size during run-time, thus adapting to network congestion. With this flow control mechanism, HyperSCSI can perform packet retransmission should a packet be lost and the expected ACK frame not be received in time. Either a selective retransmission scheme or a simpler window retransmit scheme can be used. This can be decided based on the implementation environment, thus giving users a wide degree of flexibility and performance tuning options. Figure 2 shows the different flow control mechanisms.

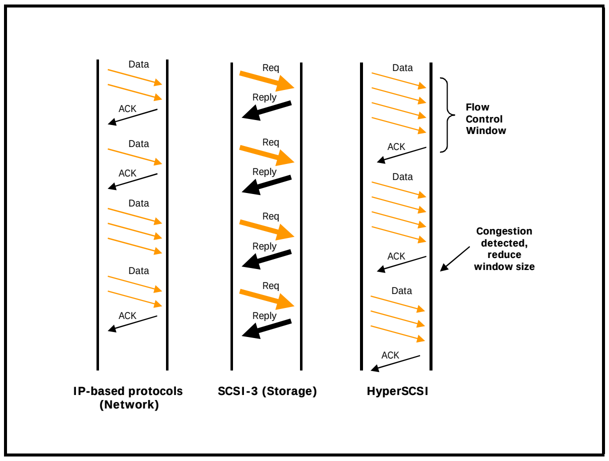

Figure 2: HyperSCSI Flow Control Compared

2.3 Security - Integrated authentication and encryption

iSCSI, iFCP and FCIP are all based on the TCP/IP network protocol. In order to provide a secure method for data transfer, they require the use of IPsec for securing the TCP/IP connection. Certainly, this is a step forward when considering that Fibre Channel's main security mechanism is LUN masking [10] which is implemented mostly on the switch. However, using IPsec adds to the complexity of the solution and implies securing the entire connection. This is different from the more flexible LUN masking method that FC uses to allow the user to secure individual LUNs as required.

HyperSCSI on the other hand has integrated security options to be specified by individual devices (or LUNs) instead of at the connection level. Of course, iSCSI for example, only supports one LUN per connection, while HyperSCSI can have multiple devices in a single connection.

During the connection initialization stage, the HyperSCSI server will authenticate the client and make a decision whether or not to export the storage resource by using the HyperSCSI Group Name and connection password. Once the authentication succeeds, a security key is also exchanged between the server and client pair. HyperSCSI allows for security to be modularized into different levels of requirements such as data hashing, encryption or none at all, thereby giving even more options to secure (or not) the device and/or the connection. By integrating the different security functions, HyperSCSI can provide more efficient, flexible and secure methods for network storage.

2.4 Reliability

As shown in Figure 3, each individual layer has its own functions to ensure the reliability of data transfer. The SCSI layer has its own functions to check and ensure that SCSI data and commands are transferred properly. TCP/IP guarantees that the network connection is reliable, while the Ethernet layer also has its own features to check data correctness.

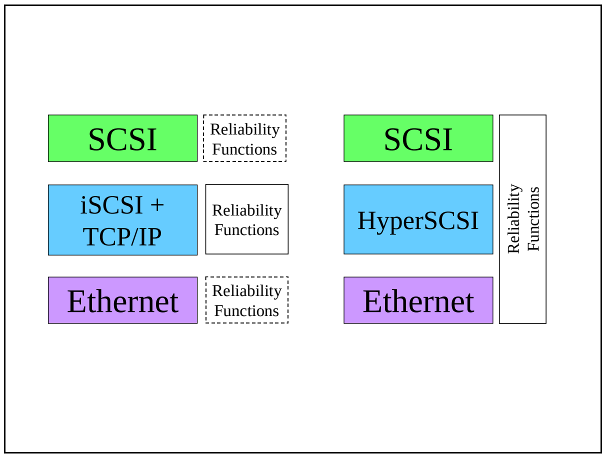

Figure 3: HyperSCSI Reliability

IP storage protocols, like iSCSI, iFCP and FCIP, rely solely on the TCP/IP layer for reliable data transfer. However, HyperSCSI makes use of the SCSI layer to ensure that SCSI data and commands are transferred properly and the Ethernet layer to check packet correctness. In addition, HyperSCSI provides its own reliable control mechanisms, such as flow control and packet retransmission. Hence, by combining the features of different layers, the entire system with HyperSCSI is as reliable as one using TCP/IP based encapsulation. However, HyperSCSI is likely to be more efficient by working with functions that already exist in other layers.

3 Benchmarking Tests and Performance Analysis of HyperSCSI

Based on our designs and protocol specifications [11], we implemented the HyperSCSI protocol on the Linux platform to see if our designs work as intended. While the reliable transfer of data over HyperSCSI is certainly verifiable, we are also very much concerned if it will perform as designed. As such, we ran a battery of tests and benchmarks over both Fast and Gigabit Ethernet to see if it can respond to the challenge. The results so far prove to be most encouraging.

3.1 Description of the test environment

For our test environment, we set up a HyperSCSI client and a server (or often called initiator and target in the storage world) connected by a network switch. Both the client and server run RedHat Linux 7.1 using the standard Linux kernel version 2.4.16 with the Linux Second Extended Filesystem (ext2). The HyperSCSI server contains an Adaptec 39160 Ultra160 SCSI controller attached to eight Seagate ST318406 LC Cheetah 18GB 10000 RPM SCSI drives. The version of HyperSCSI code we ran was 20020725. We used the hdparm, dd, file copy (cp), iozone, bonnie++ and sar programs as benchmarking tools. The Iozone version used was 3.71, while the bonnie++ version was 1.02a. For tests involving copy/read from disk, the destination (copy to) used was /dev/null. Figure 4, shows a picture of our GE test environment. All our results are obtained by averaging the output from running the same test five times.

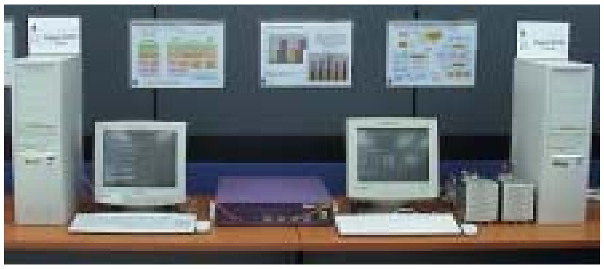

Figure 4: HyperSCSI GE Test Environment

3.2 HyperSCSI over Fast Ethernet

Our first test is to investigate if HyperSCSI can fully utilize a simple channel, like Fast Ethernet efficiently. The results of this test shows that even Fast Ethernet can provide network storage, although not at the kind of speeds that Fibre Channel is used to.

In this test, both the HyperSCSI server and client use an Intel Pentium III 1GHz CPU, 32bit 33MHz PCI bus, 256MB 133MHz SDRAM. The Network Interface Card (NIC) used is a 3Com 3C905B-TXNM card and the switch is Cisco Catalyst 3500 XL with 24 ports for 10/100 Fast Ethernet.

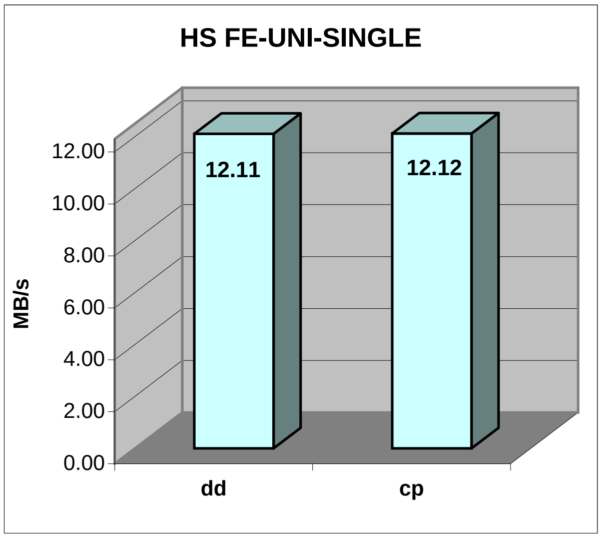

Figure 5: HyperSCSI Fast Ethernet Performance

There are two kinds of test results shown in Figure 5, dd and cp. For the dd test, the data set chosen is 1GB of raw data while for cp test, the file used is a large MPEG file of size 616137020 Bytes. By using large data sets, we hope to minimize or eliminate the effects of local cache on the test results. The maximum Ethernet frame size is 1500 Bytes. Using the HyperSCSI protocol, the user data transfer rate reaches up to 12.11 MB/sec.

In order to understand the channel utilization, we can conduct a theoretical calculation. From the Ethernet standard [12], one frame contains an 8-byte preamble, a 14-byte header, a 4-byte CRC, a 12-byte inter-frame gap and 1500 bytes of data. With the HyperSCSI protocol, every packet has a 3-byte HyperSCSI header. Thus the ratio of the bandwidth available for HyperSCSI user data is 1497 out of 1538. Considering the channel bandwidth is 100Mbit/sec, the theoretical boundary for HyperSCSI is 12.16 MB/sec.

However, in actual implementations, various overheads need to be factored in. This includes the protocol overhead, SCSI block data that may not always be fragmented into a full Ethernet frame size and so on. But as our measured results are quite close to the theoretical limits, we believe that HyperSCSI is indeed quite efficient, from both theoretical calculations and actual measurement.

3.3 HyperSCSI over Gigabit Ethernet

We then conducted our benchmark tests on Gigabit Ethernet. Both the HyperSCSI server and client use an AMD 1.2GHz Athlon dual CPU (but working in uni-processor mode), 64bit 33MHz PCI bus, 256MB 266MHz DDR RAM. The Network Interface Card (NIC) is a Syskonnect SK-9843 GE-SX and the switch is Extreme Summit 5i Model 11503 with 16 1000BaseSX port. The Ethernet frame size is 1500 bytes.

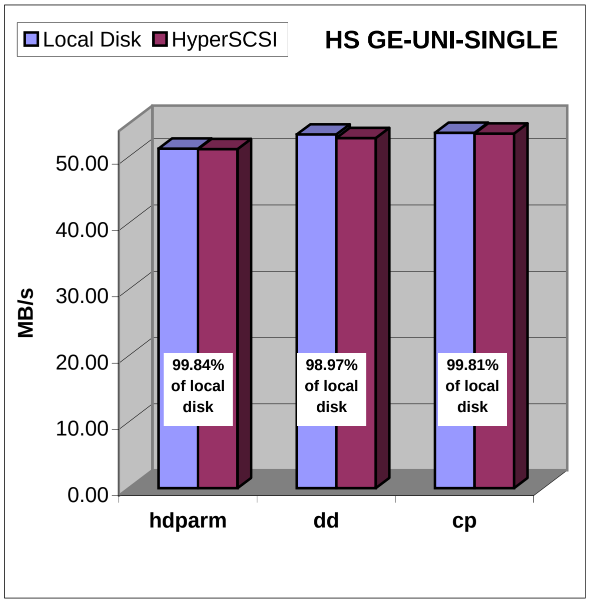

Figure 6: HyperSCSI Gigabit Ethernet Performance

From Figure 6, we can see that HyperSCSI is able to achieve almost the same performance accessing the disk over the network from the client as compared to accessing the same disk locally on the server itself. This is an interesting observation and shows that when the storage bandwidth is less than the network bandwidth, and CPU processing power is enough, the HyperSCSI client machine may not see any difference between accessing a network storage device, and its own local disk.

3.4 Top Performance of HyperSCSI with RAID and GE Jumbo Frames

In order to view the highest performance of HyperSCSI protocol, we configure the HyperSCSI server to export 8 SCSI disks to client, and build software RAID0 on the client machine. We ran the test using both normal and Jumbo frames (9000 Bytes) on Gigabit Ethernet. The result is shown in Figure 7.

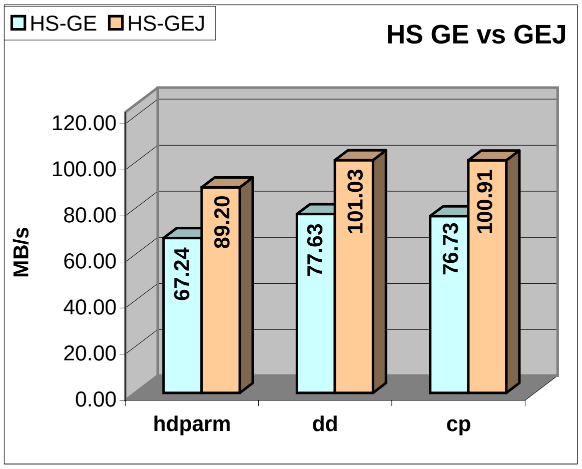

Figure 7: GE and GE Jumbo Performance

We can see that the benchmark results of both dd and cp are larger than 100MB/sec. This is significant value that indicates that existing 1 Gigabit Ethernet can achieve the same or better performance when compared to 1 Gigabit Fibre Channel. This shows that a high performance Ethernet storage network is possible without needing to resort to special hardware or software. Results from the hdparm test do not reach 100MB/s because hdparm only uses a 64MB data set, which is insufficient to measure sustained data transfer throughput at high speeds.

3.5 Performance Comparisons

Every experiment needs a control. Otherwise, it would be impossible to ascertain if results obtained were influenced by other factors. We chose to use the iSCSI and NFS protocols as a form of control for us to understand our own benchmark results. For these tests, we downloaded and compiled Intel's iSCSI version 8 code and used NFS version 2 over UDP from the standard RedHat Linux RPM version 0.3.1-5.

3.5.1 Disk Access Efficiency

Earlier in section 3.3, we measured HyperSCSI's ability to meet the same performance as the local disk. Wanting to see if this can also be done with the other protocols, we re-ran the same tests with the iSCSI and NFS.

	HyperSCSI	**iSCSI**
dd test	98.97%	79.01%

	HyperSCSI	**iSCSI**	**NFS**
cp test	99.81%	60.40%	49.25%

Table 1: Comparison to Local Disk Performance

In Table 1, we present the performance results of NFS, iSCSI and HyperSCSI as a percentage of local disk access. Naturally, NFS is a file-sharing protocol, and we do not run block-level dd tests on it. The results show that HyperSCSI is more efficient at exporting a local disk over a network than either iSCSI or NFS in the same environment. This is true for both block and file level access.

3.5.2 IRQ Comparison

A common question regarding iSCSI has been the additional overhead imposed by the TCP/IP stack. This is also manifested as the Interrupt Requests (IRQs) generated by the NICs, a point that FC HBAs, TOEs and iSCSI HBAs are also designed to address as well. While some new Ethernet NICs have advanced Interrupt Coalescing capabilities, it is still worthwhile to measure and quantify the impact of IRQs on data transfers.

	iSCSI	HyperSCSI
Total number of IRQs	199627.4	170307.2

Table 2: IRQs Generated for 1GB of Data

The results in Table 2 were obtained by using sar to measure the number of interrupts generated by the client NIC during a dd transfer of 1GB of data over GE Jumbo Frames. While MB/sec throughput numbers may tell the speed of the transfer, these results show that regardless of the time taken, iSCSI generates 17.22% more IRQs than HyperSCSI to transfer the same amount of data. This proves that HyperSCSI does reduce the IRQ overhead in a system.

3.5.3 File Access Performance Comparison

In the end, many choose to just focus on whether or not a technology (like HyperSCSI) can provide better performance in the same environment.

As can be seen from Figure 8, HyperSCSI is able to consistently provide higher performance for both reads and writes at the file system level when compared to iSCSI and NFS in the same environment.

3.6 HyperSCSI Performance in a Mixed Traffic Environment

One of the reasons to adopt an Ethernet-based network storage solution is to consolidate hardware resources. If one can use the same NIC card for both LAN and SAN access, it would represent cost savings, provided the users understood the potential impact on performance it would entail.

To assist users in making such decisions, we decided to run some tests to understand the impact and performance on a HyperSCSI client should the client also have other network traffic being sent to it. This is important because most clients are expected to be application servers of some type, accessing the SAN. As such, they will more than likely have other network communications being sent to it.

These tests were conducted using a Fluke Link Analyzer to generate fixed amounts of traffic to the client machine. The packets generated were 512-Byte frames to simulate other types of protocols and short messages. By varying the load sent by the Link Analyzer in a linear fashion (10%, 20%, 30%, onwards), we were able to gauge the impact of different network loads together with HyperSCSI in the same

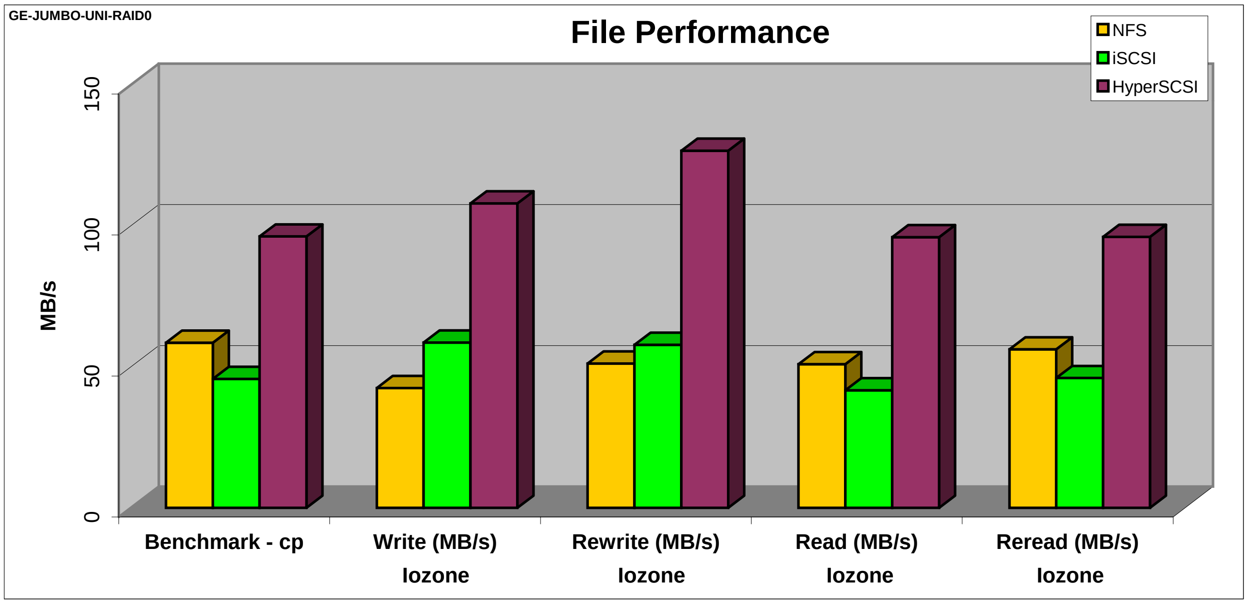

Figure 8: File Access Performance Comparison

environment. We conducted this experiment with both normal and Jumbo Frames.

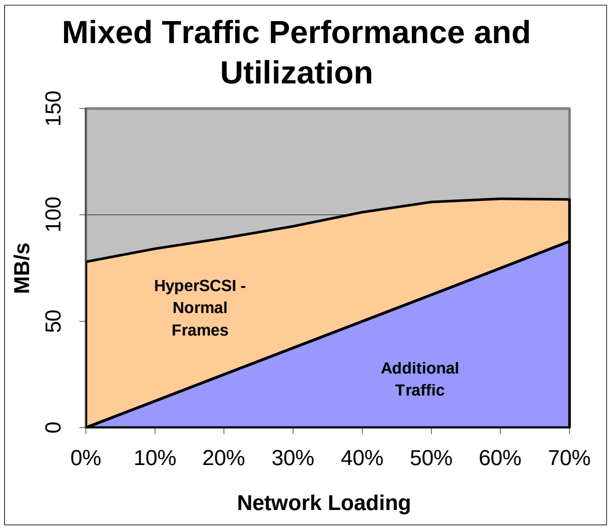

Figure 9: Mixed Traffic Utilization - Normal Frame

The results in Figure 9 show that for normal Ethernet frames, when the network load is light, HyperSCSI traffic cannot use all remaining available bandwidth. This is likely due to the Gigabit NIC processing overhead on the HyperSCSI machine, therefore the total channel utilization is not full. Only when the network load increases to a certain amount, does the total channel utilization reach close to 100%.

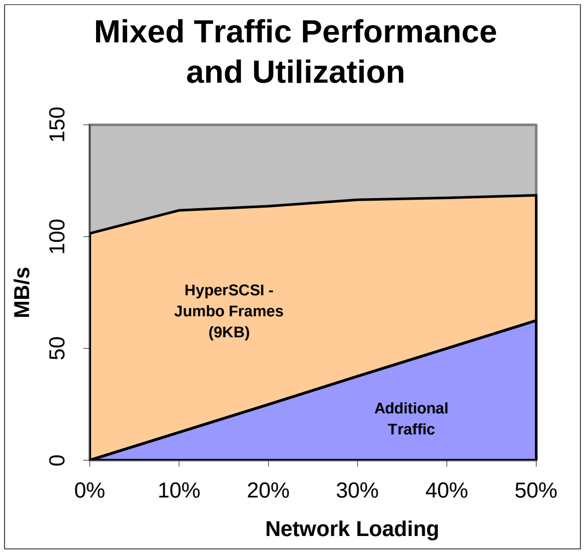

Figure 10: Mixed Traffic Utilization - Jumbo Frames

For Jumbo Frames however, the total channel utilization with HyperSCSI reaches close to 100% much faster than normal frames. Again, we suspect this is due to the NIC processing overhead, which is less for GE Jumbo than for normal frames. These results are presented in Figure 10.

On the whole, HyperSCSI can adapt to meet changing or diverse network load conditions and still provide reliable network storage data transfer.

4 Discussion and Conclusion

In the introduction of this paper, we looked back and understood that network storage is in fact not really a new idea. Neither are SCSI, Ethernet and TCP/IP. But why reminisce? Looking back in history shows us that in fact, the primary drivers for growth in both economic and technical terms has been very simply - “Can I do more for less?” If we understand this very important point, then all other predictions on trends and evaluations on technologies can be put into its proper focus.

Also in the introduction, we covered how users were not completely satisfied with Fibre Channel and iSCSI-type solutions. Whether it was complexity or cost, the adoption has just not been as strong as forecasted [13]. Do users really get more for less? The naysayers therefore, point to the possibility that Ethernet can’t really do storage.

However, in the rest of this paper, we tried to illustrate a new solution called HyperSCSI, which attempts to truly give users more for less. It is not over-engineered, is simple in concept, does not require special hardware or customized software and manages to provide a reasonable level of performance.

Can people really get more for less? We believe the answer is an unqualified yes. However, as we outlined in the beginning of this paper, doing so requires us to change our mindsets.

For example, if TCP/IP is really the bottleneck for storage over Ethernet, then why use it? Do we really need it? Of course, for certain wide-area connectivity applications, like disaster recovery and so on, a solution like iSCSI is quite a good idea [1][14]. Some long distance tests were successfully conducted, demonstrating iSCSI storage between Israel and California [15]. But in fact, most SAN implementations are not for use in wide-area long-distance applications. If it were, FC would have already died out by now since FC is itself also a local-area networking technology that needs something like FCIP to bridge the router divide. Eliminating TCP/IP would eliminate the need for hardware accelerators while still achieving high performance Ethernet-SANs. A nice side effect of eliminating TCP/IP is that the disk array providing Ethernet-SAN can't be hacked from the Internet.

Furthermore, HyperSCSI gives rise to entirely new applications and markets. For example, HyperSCSI runs quite well on wireless LAN, thus allowing the development of a wireless HDD or CDRW for your laptop instead of through USB or Firewire. Or how about watching a movie on a webpad from the patio by directly accessing the DVD player in your living room wirelessly? With such new developments, it's no wonder that there is a renewed sense of optimism for the network storage industry.

No, we believe that Ethernet storage is not coming – it's already here. And yes, you will get more for less.

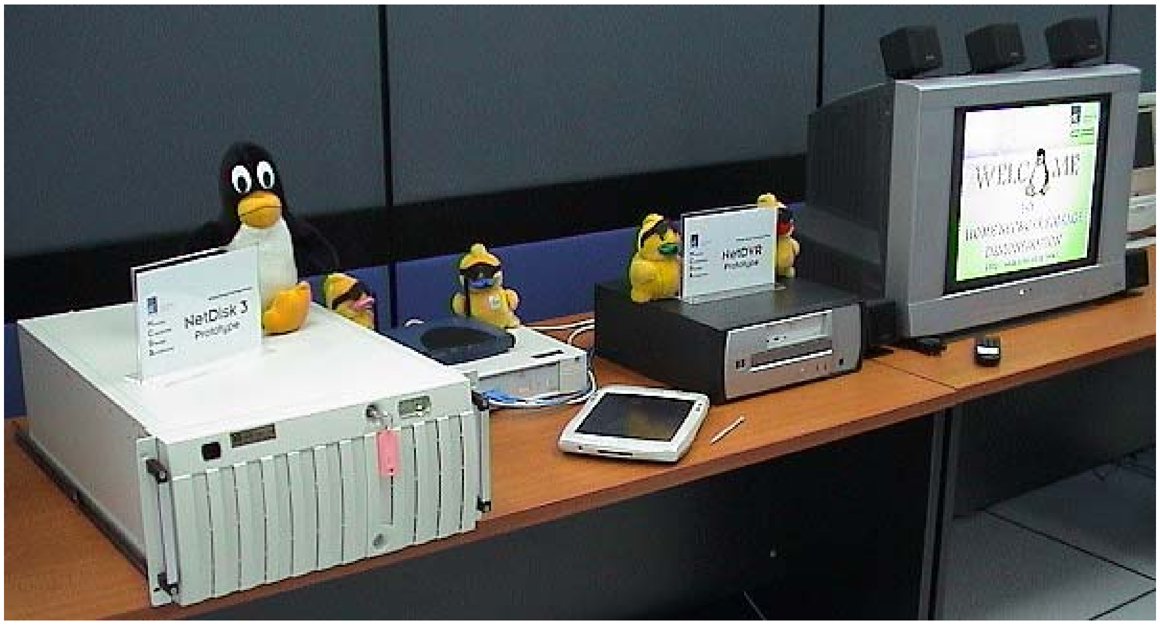

Figure 11: HyperSCSI Demonstration Environment for Ethernet-SAN and wireless storage

References

[1] Prasenjit Sarkar, Kaladhar Voruganti, "IP Storage: The Challenge Ahead", 10th Goddard Conference on Mass Storage Systems and Technologies and 19th IEEE Symposium on Mass storage System. Page(s): 35-42

[2] Robert Horst, "IP Storage and CPU Consumption Myth", 2001 IEEE International Symposium on Network Computing and Applications. Page(s): 194-200

[3] Huseyin Simitch, Chris Malakapalli, Vamsi Gunturu, "Evaluation of SCSI Over TCP/IP and SCSI Over Fibre Channel Connections", Hot Interconnects 9, 2001. Page(s): 87-91

[4] Patrick B. T. Khoo, Wilson Y. H. Wang, " Introducing A Flexible Data Transport Protocol for Network Storage Applications", 10th Goddard Conference on Mass Storage Systems and Technologies and 19th IEEE Symposium on Mass storage System. Page(s): 241-257

[5] IPS Working Group Internet Engineering Task Force, "iSCSI Internet Draft", September 2002

[6] IPS Working Group Internet Engineering Task Force, "Fibre Channel over TCP/IP (FCIP) Internet Draft", August 2002

[7] Jeffrey S. Chase, Andrew J. Gallatin and Kenneth G. Yocum, "End System Optimisations for High-Speed TCP", IEEE Communications Magazine, April 2001

[8] HyperSCSI Project Home Page, http://nst.dsi.a-star.edu.sg/mcsa/hyperscsi/

[9] IPS Working Group Internet Engineering Task Force, "Internet Storage Name Service Internet Draft", August 2002

[10] John Vacca, "The Basics of SAN Security - Part II", Enterprise Storage Forum, July 25, 2002, http://www.enterprisestorageforum.com/sans/features/article

[11] "HyperSCSI Protocol Specification", MCSA, Data Storage Institute, Singapore. http://nst.dsi.a-star.edu.sg/mcsa/hyperscsi/docs.html

[12] Mitchell L. Loeb, et.al, "Gigabit Ethernet PCI Adapter Performance", IEEE Network, Volume: 15 Issue: 2, March-April 2001 Page(s): 42-47

[13] Todd Spangler, "iSCSI in Exile", Byte and Switch news, Aug. 14, 2002. http://www.byteandswitch.com

[14] Roy Levine, "IP-based Storage: Benefits and Challenges", Infostor, March 2001

[15] "First successful continent-to-continent iSCSI demonstration", IIS news, Feb. 12, 2002. http://www.iislf.com/