# Storage Performance over Fast Ethernet, Giga Ethernet, Giga Ethernet TOE and InfiniBand

*Omar Aldaoud*[*]*, Houssain Kettani*[*]*, Krishnapriya Guduru*[*] *and Qutaibah Malluhi*[**]

* Department of Computer Science
Jackson State University
Jackson, MS 39217

** Department of Computer Science and Engineering
Qatar University
Doha, Qatar

## Abstract

The performance of network storage systems is often limited by overheads in the I/O path, such as memory copying, network access costs, and protocol overhead. As a result, I/O remains the bottleneck for high-performance data-intensive applications. This paper examines different communication transports and their effect on network storage system's performance. To accomplish the proposed study, we have utilized a parallel network storage system and different communication transports like Fast Ethernet, Gigabit Ethernet, TOE and IP over InfiniBand (IPoIB). The purpose of this work is to evaluate the performance of parallel storage systems over the aforementioned transport access protocols.

**Keywords**: Parallel, Storage, Performance, Gigabit, TOE, IPoIB, InfiniBand

## 1. Introduction

Network storage systems have emerged as an important research field that is driven by the demand for high-capacity, high-performance, and highly available fault-tolerant storage structures. These systems were designed to satisfy the tremendous growth in information storage induced by a new class of data-intensive applications. In addition, storage capacities and computational speeds have achieved tremendous growth. In the last few years, we have seen a continuous improvement in the performance of networks, both in reduced latency and in increased bandwidth. These improvements have motivated the interest in the development of network storage systems that can take advantage of these improvements.

TCP/IP is the most popular communication protocol for Ethernet. However, the reliability provided by TCP/IP has a price in communication overhead. To ensure reliable data transfer, protocol stack implementations like TCP/IP usually require data to be copied several times among system layers and the communicating nodes exchange numerous protocol-related messages during the course of data transmission and reception. The number of protocol layers that are traversed, data copies and context switches directly contributes to the software overhead. Several studies have shown that the TCP/IP processing cause high per-byte overhead, leaving a high performance network under-utilized especially for large data movement, see [2] for more details.

Thus, the performance of network storage systems is often limited by overheads in the I/O path, such as memory copying, network access costs, and protocol overhead. As a result, I/O remains the bottleneck for high-performance data-intensive applications. Hence, modern high-performance computing systems require storage systems capable of storing huge amounts of data that is delivered to computing elements at high speeds.

Sarkar et al. [5, 6] evaluated storage protocols using Software approach, as well as two hardware approaches: TCP Offloaded Engine (TOE) and Host Bus Adapter (HBA), where the former offloads TCP/IP processing to the network interface and the latter offloads the whole storage protocol. Radkov et al. [4] compared the performance between NFS and iSCSI for IP networked storage and showed iSCSI based storage protocol's performance benefits from its better aggregation and caching capability. Both efforts target IP-based network storage.

In this paper, we examine the performance of Network Storage Manager (NSM) [1] over different communication transports that use TCP/IP protocol for communication. The paper demonstrates how NSM

system can be adapted to utilize new communication transport technologies to leverage performance while providing the reliability and fault tolerance features required by storage systems.

The layout of this paper is as follows. Section 2 outlines the features of NSM, Fast Ethernet, Gigabit Ethernet, TCP/IP Offloading Engines (TOE) and InfiniBand. Section 3 provides empirical analysis that illustrates the experimental comparison of NSM over different protocol stacks and their effect on the system performance. Finally, our conclusion is in Section 4.

## 2. Preliminaries

In this section, we provide an overview of the technologies utilized in our study, mainly, NSM, Fast Ethernet, Giga Ethernet, TOE Giga Ethernet and InfiniBand. We then show how NSM reads data through different transport access protocols from remote servers.

### 2.1 Network Storage Manager

Network Storage Manager (NSM) was developed in the Distributed Computing Laboratory at Jackson State University [1]. It has been used to explore I/O performance improvement for network storage systems and consolidate the physical storage of multiple hosts. NSM is a robust, scalable, high-performance, platform-independent, distributed mass storage system. It is used as a data storage and access framework for data intensive applications. Through its unique, pluggable architecture, NSM offers its applications the ability to control the behavior of data storage, access environments and transport mechanism. Consequently, NSM enables an application to tune and enhance its I/O performance.

As illustrated in Figure 1, NSM stripes its files across multiple distributed storage servers. Accordingly, concurrent data streams to and from the parallel servers are used to achieve high data rates and perfect load balancing. In addition, coded redundancy is added to the data and stored on distinct parity storage servers. This redundancy is used to effectively recover from transient and permanent server and data faults to achieve reliability and high-availability. Hence, the system offers a self-healing feature by automatically replacing permanently failing servers with operational backup servers.

NSM generates and manages storage meta-data, which maintains information about the structure of the file and the location of its components. Consequently, the system utilizes the meta-data to provide location transparency to users and allows them to deal with remote shared data objects with the same ease as

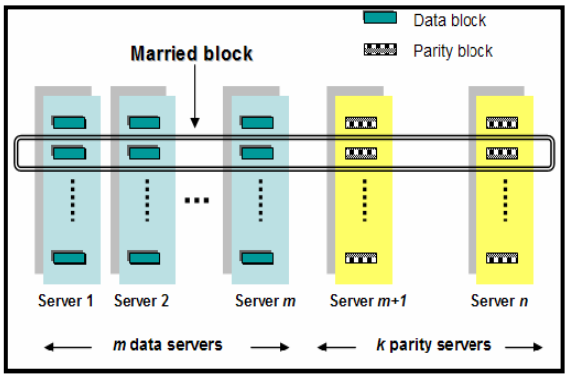


**Figure 1** - NSM Data Layout over Storage Nodes

dealing with a local file. This allows data to become more widely available to geographically distributed users. Thus, there is no need to distribute copies of files.

Several mechanisms can be used for reading and writing remote data. For example, FTP is one of the most common network data access protocols. However, many application environments may not find it optimal for many reasons such as performance and security. FTP may be preferred for data distribution but not for retrieval. NSM provides flexibility in choosing the data access protocol. In NSM, applications have full control over the used network protocol. An application can plug-in its own protocol by implementing the protocol interface provided by NSM.

### 2.2 Fast Ethernet

Fast Ethernet is a collective term for a number of Ethernet standards that carry traffic at the nominal rate of 100Mbit/s, against the original Ethernet speed of 10Mbit/s. Introduced in 1995, Fast Ethernet is no longer the fastest form of Ethernet: Gigabit Ethernet and the new 10 Gigabit Ethernet standards are 10 and 100 times faster, respectively.

### 2.3 Gigabit Ethernet

Introduced in 1999, the Gigabit Ethernet [8] is a high-speed form of Ethernet that can provide data transfer rates of about 1 gigabit per second (Gbps). Gigabit Ethernet is an extension of the highly successful 10Mbps (10BASE-T) Ethernet and 100Mbps (100BASE-T) Fast Ethernet standards for network connectivity. In short, Gigabit Ethernet is the same Ethernet but 10 times faster than Fast Ethernet and 100 times faster than the classical Ethernet. It supports additional features that accommodate today's bandwidth-hungry applications and match the increasing power of servers and desktops.

### 2.4 TCP Offloading Engine

TCP Offloading Engine (TOE) implementation was first developed and patented in 2004 by Valentin Ossman [9]. TOE technology aims to take the server CPU out of I/O processing by shifting TCP/IP processing tasks to the network adapter or storage device. This leaves the CPU free to run its applications, so users get their data faster. TOE components are incorporated into one of the printed circuit boards, such as Network Interface Card (NIC) or Host Bus Adapter (HBA).

### 2.5 IP over InfiniBand

The first version of InfiniBand specification was completed in 2000 by the InfiniBand Trade Association (IBTA), which was founded in 1999 [7]. InfiniBand (IB) is utilized as a mechanism that provides direct access and hardware transport protocol features. InfiniBand Architecture (IBA) is an industry standard that defines a new high-speed switched fabric subsystem designed to connect processor nodes and I/O nodes to form a system area network. IP over InfiniBand (IPoIB) is an IP emulation for InfiniBand, which enables IP based application to run over InfiniBand without any modification.

## 3. Performance Evaluation

In order to measure the effect of data access protocol on NSM, we have performed several experiments utilizing the NSM prototype for reading data over different data access protocols with varying parameters. The main intention was to allow an evaluation of the different data access protocols effect on the throughput of NSM. Therefore, this section presents the performance comparison results of NSM using different data access protocols such as Fast Ethernet, Gigabit Ethernet, Gigabit TOE and IPoIB utilizing our test bed.

We start by presenting our experimental test bed that was used for the evaluation of the system. We then demonstrate that NSM can utilize parallelism with communication transports to achieve high throughput. Thus, a series of experiments were conducted to show the effect of parallel read with different transports. The purpose of these experiments is to examine the system performance in terms of throughput, and CPU utilization for reading datasets. The experiments demonstrate the effect of caching and the number of servers used for distributing and retrieving large datasets.

### 3.1 Experimental Setup

The results that follow were gathered using the following test bed: the server nodes used in the experiments were equipped with an Intel E7501 chipset, two Intel Xeon 2.4 GHz P4 processors, 512KB L2 cache, 400 MHz front side bus, 1GB DDR SDRAM and 64-bit 133MHz PCI-X interfaces. The client nodes used in the experiments were equipped with an Intel E7501 chipset, Intel Xeon 2.4GHz P4 processor, 512KB L2 cache, 400 MHz front side bus, 1GB DDR SDRAM and 64-bit 133 MHz PCI-X interfaces each node was also equipped with a 250GB, 7200K RPM Seagate disk. Both the client and server nodes ran the Linux-2.4.21 kernel. For the IB-based implementations, we used Mellanox InfiniHost MT23108 Dual Port 4x HCA adapters connected through an InfiniScale MTS2400 24-Ports IB switch. For the Mellanox InfiniHost HCA, we used firmware version fw-08-3.3.2. All the performances reported in this paper are based on the average of 10 measurements.

Figure 2 illustrates the system stack that was used in the evaluation process. Accordingly, NSM reader application utilizes FTP to read data from remote servers through different data access protocols (FastEthernet, Gigabit, Gigabit TOE and IPoIB).

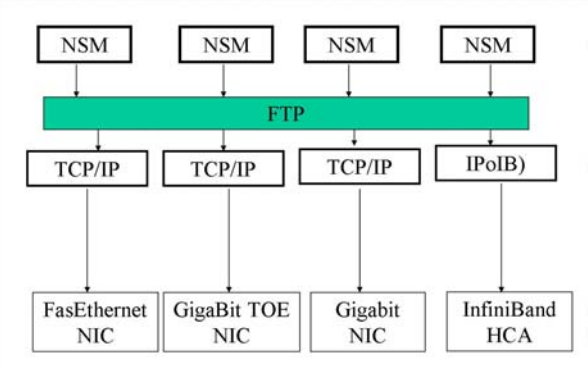


**Figure 2 –** Different data access protocol diagram

### 3.2 Performance Results

The first set of experiments investigates the effect of the number of servers and caching on the overall read throughput of NSM using Fast Ethernet, Giga Ethernet, TOE Giga Ethernet and IPoIB as the underlying transport layer and interconnects. The data was gathered using a 250MB file and a 1MB block size which has been striped among the server nodes using NSM. The file is first distributed each time using different number of server nodes: one, two, three and four servers. The file is then read entirely using NSM in parallel and the elapsed time it takes for the file to be read is recorded. The throughput (MB/s) is calculated by dividing the file size (MB) by the elapsed time (seconds) required to complete reading the entire file. Each experimental result represents the average of

ten runs. The experiments were conducted using cache (warm disk). At the beginning, the file is read once; this allows the server file system to cache the file into its memory. Next the file is read and the elapsed time is recorded. Therefore, this experiment measures only the network transfer speed that can be achieved using NSM over different communication transports.

Figure 3 depicts the results of using NSM over Fast Ethernet as the transport access protocol. From this figure, we can see that when Fast Ethernet as the data access protocol and using one server we get an average throughput of about 9.27MB/s. When using two servers, on the other hand, this average throughput increases to about 18MB/s, resulting in about 100% increase from the result of only one server. Alternatively, when using three servers, the average throughput increases to about 26MB/s, resulting in an increase from the previous case of about 44% but three times the one server. Finally, with four servers, we reach about 32MB/s, which is about 3.5 times that when using one server. Therefore the performance increase is almost linear.

Figure 3 depicts the results of using NSM over Gigabit Ethernet as transport access protocol. From this figure, we can see that when Gigabit Ethernet as the data access protocol and using one server we get an average throughput of about 47MB/s. When using two servers, on the other hand, this average throughput increases to about 52MB/s, resulting in about 2.3% increase from the result of only one server. Alternatively, when using three servers, the average throughput increases to about 63MB/s, resulting in an increase from the previous case of about 6% but 34% increase from one server. Finally, with four servers, we reach about 72MB/s which is about 53% more that when using one server.

The third set of experiments involves reading the same file using the same parameters as in the previous set of experiments but with using IPoIB as the communication transport. Figure 3 shows the throughput achieved by the system. Note that the system achieves throughput of 22MB/s, 47MB/s, 63MB/s and 90MB/s using one, two, three and four servers respectively. This increase is due to parallel read feature that NSM employs. Thus, when using one server we get an average throughput of about 22MB/s. When using two servers, on the other hand, this average throughput increases to about 47MB/s, resulting in about 113% increase from the result of only one server. Alternatively, when using three servers, the average throughput increases to about 63MB/s, resulting in an increase of about 3 times that when using one server. Finally, with four servers, we reach about 90MB/s which is about 4 times that of one server. Therefore the performance increase is linear in this case.

The last set of experiments involves reading the same file using the same parameters as in the other set of experiments but with using TOE Gigabit Ethernet as the communication transport. Figure 3 shows the throughput achieved by the system. Note that the system achieves throughput of 67MB/s, 80MB/s using one, two respectively. But in contrast system throughput is 10MB and 5MB when using three and four servers respectively.

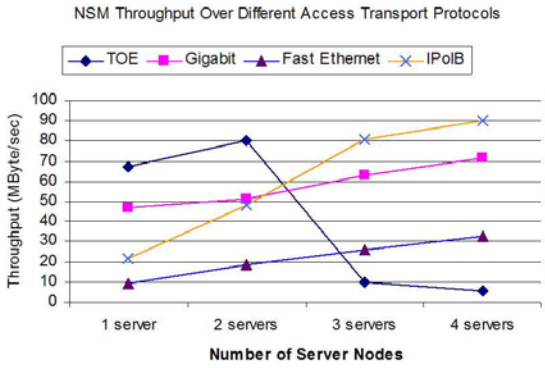


**Figure 3** – Throughput Using Different Access Transport Protocols

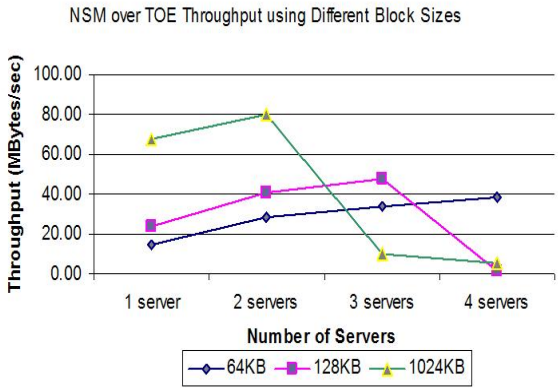


**Figure 4** – Throughput Using TOE and Variable Block Sizes

The increase from using one server to using two is due to parallel read feature that NSM employs. Thus, when using one server we get an average throughput of about 67MB/s. When using two servers, on the other hand, this average throughput increases to about 80MB/s, resulting in about 19% increase from the result of only one server. Alternatively, when using three servers, the average throughput decrease to about 10MB/s, resulting in about 19% increase from the result of only one server. Alternatively, when using three servers, the average throughput decrease to about 10MB/s, resulting in a decrease from the previous case

of about 87%. Finally, with four servers, we got about 5MB/sec with a decrease from the previous case of about 50%. This decrease in throughput is due mainly to the way the TOE process the TCP/IP protocol. Further investigation (Figure 4) of this we found that with smaller block size of 128KB the system exhibit an increase of throughput when using TOE and three servers but the throughput decreases when using four servers and 128KB block size. Consequently with a block size of 64KB the system performed as the rest of the other communication transports.

## 4. Conclusion

The results presented in this paper show the performance effect of different data access protocol transports has on storage system. This paper presented the performance of storage system utilizing different data access protocols such as Fast Ethernet, GigBit Ethernet, TOE, and IPoIB. The results demonstrate that these protocols benefit storage system with the combination when utilizing parallelism for.

Our results, generated by running our experimental methodology shows that using TOE significantly outperforms Fast Ethernet, Gigabit Ethernet and IPoIB when using one and two data servers. On the other hand the performance of Fast Ethernet, Gigabit Ethernet and IPoIB is affected by TCP/IP protocol, which uses a bit of CPU time on both server and client, thus lowering the throughput, but TOE increases the performance of TCP/IP due to offloading the TCP/IP processing to the Network Interface Card instead of the CPU.

## Acknowledgements

This work was funded by the Department of Defense (DoD) through the Engineering Research Development Center (ERDC), Vicksburg, Mississippi, under contract number W912HZ-05-C-0051

## References

[1] Z. Ali and Q. Malluhi, "NSM - A Distributed Storage Architecture for Data Intensive Applications," proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies, San Diego, California, April 2003.

[2] N. Boden, D. Cohen, R. Felderman, A. Kulawik, C. Seitz, J. Seizovic, and W. Su. Myrinet: A Gigabit-per-Second Local Area Network. *IEEE Micro*, 15(1):29–36, February 1995.

[3] InfiniBand Trade Association, "About InfiniBand Trade Association: An InfiniBand Technology Overview," http://www.infinibandta.org/ibta/.

[4] P. Radkov, L. Yin, P. Goyal, P. Sarkar, and P. Shenoy. A Performance Comparison of NFS and iSCSI for IP-Networked Storage. In *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*, San Francisco, CA March 2004.

[5] P. Sarkar, S. Uttamchandani, and K. Voruganti. Storage Over IP: When does Hardware Support Help? In *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, San Francisco, CA, January 2003.

[6] K. Voruganti and P. Sarkar. An Analysis of Three Gigabit Networking Protocols for Stroage Area Networks. In *Proceedings of International Conference on Performance, Computing, and Communications*, Phoenix, AZ, April 2001.

[7] Infiniband Trade Association. http://www.infinibandta.org.

[8] H. Frazier and H. Johnson. Gigabit Ethernet: From 100 to 1000Mbps. *IEEE Internet Computing*, Vol:3, Issue:1:24–31, January 1999.

[9] Andy Currid, iReady. TCP Offload to the Rescue, ACM Queue vol. 2, no. 3 - May 2004