pNFS/PVFS2 over InfiniBand: Early Experiences *

Lei Chai Xiangyong Ouyang Ranjit Noronha Dhabaleswar K. Panda

Department of Computer Science and Engineering
The Ohio State University
{chail, ouyangx, noronha, panda}@cse.ohio-state.edu

ABSTRACT

The computing power of clusters has been rapidly growing up towards petascale capability, which requires petascale I/O systems to provide data in a sustained high-throughput manner. Network File System (NFS), a ubiquitous standard used in most existing clusters, has shown performance bottleneck associated with the single server model. pNFS, a parallel version of NFS, has been proposed in this context to eliminate the performance bottleneck while maintain the ease of management and interoperability features of NFS. With InfiniBand being one of the most popular high speed networks for clusters, whether pNFS can pick up the advantages of InfiniBand is an interesting and important question. It is also important to quantify and understand the potential benefits of using pNFS compared with the single server NFS, and the possible overhead associated with pNFS. However, since pNFS is relatively new, few such study has been carried out in an InfiniBand cluster environment. In this paper we have designed and carried out a set of experiments to study the performance and scalability of pNFS, using PVFS2 as the backend file system. The aim is to understand the characteristics of pNFS, and its feasibility as the parallel file system solution for clusters. From our experimental results we observer that pNFS can take advantages of high speed networks such as InfiniBand, and achieve up to 480% improvement in throughput compared with using GigE as the transport. pNFS can eliminate the single server bottleneck associated with NFS. pNFS/PVFS2 shows significantly higher throughput and better scalability compared with NFS/PVFS2. pNFS/PVFS2 achieves peak write throughput about 490MB/s, and read throughput about 2250MB/s, with 4 I/O servers. With 8 I/O servers, the numbers are 754MB/s and 3100MB/s. Further, we find that pNFS adds little overhead and achieves almost the same throughput as the backend file system PVFS2. Our results indicate that pNFS is promising as the parallel file system solution for clusters.

Categories and Subject Descriptors

D.4.3 [**File Systems Management**]: Distributed file systems; D.4.4 [**Communications Management**]: Input/output, Network communication; D.4.8 [**Performance**]: Measurements

General Terms

Network based file systems performance measurements

Keywords

pNFS, PVFS2, NFS, InfiniBand, parallel file system, cluster, networking

1. INTRODUCTION

Commodity clusters are a popular cost effective platform for both High Performance Computing (HPC) and datacenters. Large scale clusters running long running scientific applications use and generate terabytes and in some cases petabytes of data. Satellite imagery, oceanography and a variety of other fields generate petabytes of data, which is must be stored and accessed in a efficient manner.

Network File System (NFS) is currently being used as a ubiquitous standard in most clusters. It has several advantages and one of the most important is that it is an open standard and any vendors can pick up and have their own implementations that are interoperable with others. While served sufficiently well in the past, NFS has revealed performance problem as the size of clusters scales. The main reason is that NFS is a single server model which becomes a bottleneck in a large scale cluster. In this situation, researchers have proposed parallel file systems that decouple the data and metadata paths and distribute data to multiple storage servers and allows clients to access storage servers in parallel, such as PVFS2 [4], Lustre [3], etc. These parallel file systems have shown very good performance. However, the lack of a standard protocol makes interoperability an issue. And the clients often need to be reinstalled when deploying a new type of back-end file system on the server.

pNFS has been proposed to bridge these gaps. It is an extension to NFS that allows clients to access multiple storage servers directly and in a parallel manner thus eliminating the single server bottleneck. Since it is still NFS, it facilitates interoperability. pNFS currently supports three types of data layouts - blocks, files, and objects. The pNFS clients can essentially access different types of back-end file systems in a transparent manner. And when a new type of file system emerges, the clients will just need to install the layout driver for the new file system. Previous work has focused on pNFS performance using a PVFS2 layout driver [6, 8, 7]. where PVFS2 used TCP over Gigabit Ethernet as the underlying transport. The limited bandwidth of Gigabit Ethernet in concert with the state and copying overhead of multiple TCP connections imposed natural bounds on the stripping width and scalability of large horizontally scaled parallel file systems. With InfiniBand [1] being one of the most popular high performance networks for clusters and pNFS proposed as the next generation parallel file system solution for HPC, it is important to have a comprehensive study on pNFS performance over InfiniBand. In this paper we evaluate pNFS with a PVFS2 layout driver; where PVFS2 uses InfiniBand as the underlying transport fabric. To the best of our knowledge this is the first such study in the literature. Specifically, we want to answer these questions:

- What are the advantages of using InfiniBand over Gigabit Ethernet in a parallel file system environment?
- How much is the performance gain if we use pNFS instead of the traditional single server NFS?
- Is there overhead introduced by the pNFS PVFS2 layout driver compared to a native PVFS2 installation?
- How does pNFS scale with increasing number of I/O servers?

Our experimental results show that when pNFS is used through PVFS2 over InfiniBand, write throughput can be increased by up to 190% and read throughput can be increased by up to 480%, compared with a similar setup of pNFS that uses Gigabit Ethernet. Compared with the traditional NFS, pNFS through PVFS2 provides significantly higher throughput and shows better scalability. pNFS/PVFS2 achieves peak write throughput about 490MB/s, and read throughput about 2250MB/s, with 4 I/O servers. The write and read throughput with 8 I/O servers is 754MB/s and 3100MB/s respectively. It adds very little overhead and achieves almost the same throughput as the native PVFS2. It also scales well with increasing number of I/O servers. From our experience, we believe that pNFS on InfiniBand clusters is promising.

The rest of the paper is organized as the following: Section 2 discusses the background of pNFS, PVFS2, and InfiniBand. The experiment architecture is illustrated in Section 3. Section 4 presents our preliminary experimental results on a single-core Intel Xeon cluster and Section 5 presents our experimental results on a quad-core Intel Clovertown cluster. We discuss about the related works in Section 6. We conclude and and point our future directions in Section 7. And finally Section 8 is the acknowledgements.

2. BACKGROUND

In this section we discuss the background of pNFS, PVFS2, and InfiniBand.

Parallel NFS (pNFS) is an extension to NFSv4 that separates metadata and data paths, and allows clients to access storage devices directly and in parallel. pNFS has been proposed to eliminate the single server bottleneck associated with the current NFS servers while keep the ease of management and interoperability features of NFS. pNFS is being standardized by IETF [9]. pNFS data operation protocol supports three storage layouts - blocks, files, and objects, and may be extended to other layouts in the future. There are pNFS projects being developed on both Linux and Solaris [12] operating systems. In this paper we focus on the Linux implementation by the CITI group at University of Michigan [5].

Parallel Virtual File System version 2 (PVFS2) [4] is a high performance parallel file system designed for clusters. PVFS2 provides multiple interfaces, including PVFS2 specialized interface, VFS interface through a Linux kernel module, and MPI-IO via ROMIO. PVFS2 supports multiple networks as the transport, such as Ethernet, Myrinet, and InfiniBand.

The InfiniBand Architecture [1] (IBA) defines a switched network fabric for interconnecting processing and I/O nodes. It provides both Send/Receive and RDMA capabilities. In an InfiniBand network, hosts are connected to the fabric by Host Channel Adapters (HCAs). A queue based model is used in InfiniBand. A Queue Pair (QP) consists of a send and a receive queue. Communication operations are described in the Work Queue Requests (WQR), or descriptors, and submitted to the work queue. It is a requirement that all communication buffers be posted into receive work queues before any message can be placed into them. In addition, all communication buffers need to be registered (locked in physical memory) before any operations can be issued from there. This is to ensure that memory is present when HCA accesses the memory. Finally, the completion of WQRs is reported through Completion Queues (CQ). InfiniBand provides several types of transport services: Reliable Connection (RC), Unreliable Connection (UC), Reliable Datagram (RD) and Unreliable Datagram (RD). RC is the most widely used mode due to its rich features.

InfiniBand has been widely deployed in clusters to achieve low latency and high throughput for communication among nodes. The InfiniBand standard supports single data rate (SDR), double data rate (DDR), and quad data rate (QDR), which provides bandwidth equal to 10Gbps, 20Gbps, and 40Gbps, respectively.

3. DESIGN OF EXPERIMENTS

To answer the questions brought forward in Section 1, We have designed the following experiments as shown in Figure 1:

1. pNFS with a PVFS2 layout driver (a. pNFS/PVFS2)
2. PVFS2 with a VFS mount (b. PVFS2), and

3. NFS server using a PVFS2 file system as the backend (c. NFS/PVFS2)

In these configurations, InfiniBand (IB) or Gigabit Ethernet (GigE) is used as the network transport for PVFS2. For simplicity we only show one client in Figure 1.

We analyze the results using the following approach:

1. We compare the performance of using InfiniBand and GigE as the transport in the same configuration, so that we can quantify the benefits of using high speed networks in a parallel file system environment.
2. By comparing the performance of pNFS/PVFS2 and NFS/PVFS2, we can see if the performance of single server NFS can be improved by deploying pNFS. And if yes, how much improvement we can achieve.
3. We also compare the performance of pNFS/PVFS2 with that of native PVFS2, and find out the overhead associated with pNFS, if any.
4. Finally we vary the number of I/O servers, and evaluate the scalability of pNFS with increasing number of I/O servers.

The results of the comparisons are shown in the next two sections.

4. EXPERIMENTAL RESULTS ON AN INTEL XEON CLUSTER

In this section, we present and discuss our results on an Intel Xeon cluster.

Experimental Setup: Each node has dual 2.66GHz Intel Xeon processor and 2GB main memory, and is equipped with a Mellanox MT23108 InfiniBand HCA (SDR) on a 133 MHz PCI-X bus. The nodes run Linux kernel 2.6.17 with pNFS support. All nodes use OpenFabrics stack OFED 1.2.

The PVFS2 setup consists of 1 node as the metadata server and 4 nodes as I/O servers. The metadata server exports this PVFS2 file system through pNFS and NFS respectively. 4 nodes are used as clients.

We run IOzone [2] with clustering mode in this cluster to measure the aggregated write/read throughput. All IOzone test threads are evenly distributed across the client nodes. We let PVFS2 stripe size be 2MB, IOzone record size be 512KB, and IOzone file size be 64MB.

4.1 Impact of InfiniBand on pNFS/PVFS2 Performance

In the first experiment, we compare the performance of pNFS/PVFS2 on InfiniBand with GigE. Figure 2(a) and Figure 2(b) show that pNFS achieves a better throughput on InfiniBand compared with that on GigE. InfiniBand is about 10% better than GigE in terms of write in this experimental configuration. In terms of read performance, InfiniBand outperforms GigE by about 1.7 times at 16 client threads.

4.2 Comparison of pNFS, PVFS2 and NFS

In this test we measure performance of PVFS2, pNFS/PVFS2 and NFS/PVFS2 running on InfiniBand transport. We used a disk based and memory based back-end file systems at the storage nodes. Figure 3(a) shows the write performance results, and Figure 3(b) gives read performance results. These results show that pNFS closely matches the performance of PVFS2, and scales up with the back-end PVFS2. We also see that pNFS achieves much better performance than NFS. In NFS, all data I/O has to concentrate at the single metadata server before data are sent to client threads, which makes the metadata server a bottleneck in the system. On the contrary, pNFS enables a client to directly fetch data from data servers, thus effectively eliminating the single server bottleneck in NFS. pNFS may potentially scale up well with the back-end PVFS2 file system.

This test also shows that ramfs-based storage largely outperforms disk-based storage in write performance test. In disk-based storage the back-end PVFS2 writes data directly to disk, so disk speed becomes the highest constraint in write throughput. On the other hand, ramfs-based storage stores data in memory instead of writing them to disk. Its performance is only subject to network performance, network protocol stack overhead etc. instead of disk speed. When it comes to read, disk-based storage achieves a similar performance to ramfs-based storage. After write test, data is temporarily buffered in memory in disk-based storage, and immediate following read test will get data directly from memory of the storage nodes. So the situation is basically the same in disk-based storage as in ramfs-based storage in this test scenario. Again we notice that pNFS achieves a scalable performance that is very close to the backend PVFS2.

5. EXPERIMENTAL RESULTS ON AN INTEL CLOVERTOWN CLUSTER

In this section, we migrate the similar experiments from the previous section to a larger scale Intel Clovertown cluster with more advanced hardware configurations.

Experimental Setup: This cluster consists of 64 compute nodes and 8 storage nodes. Each node is equipped with dual quad-core Intel Clovertown processors, which are 8 cores at 2.33GHz per node. Each node has 6 GB main memory with PCI-Express bus. These nodes are interconnected with both Gigabit Ethernet and Mellanox InfiniBand DDR HCAs. For the 8 storage nodes, each of them is equipped with RAID disks with read/write bandwidth at 600MB/s. Software configuration is similar to that described in the previous section that each node runs Linux 2.6.17 kernel with pNFS support and uses OpenFabrics stack OFED 1.2.

The backend PVFS2 file system consits of 1 compute node as metadata server, and I/O servers are located on the storage nodes. The RAID systems on storage nodes are used as storage subsystem for PVFS2.

We run IOzone with clutering mode to collect the aggragate write/read performance data in this cluster. Each IOzone thread runs on a separate compute node. PVFS2 stripe size is set to be 4MB, IOzone record size be 2MB and IOzone

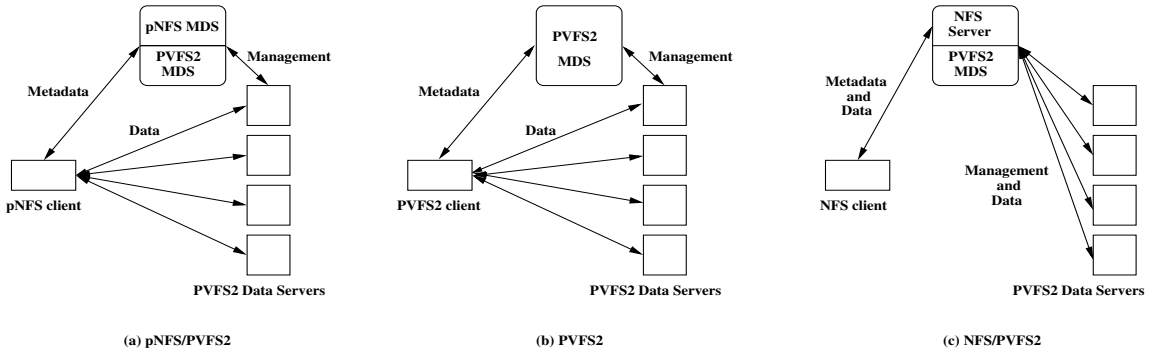
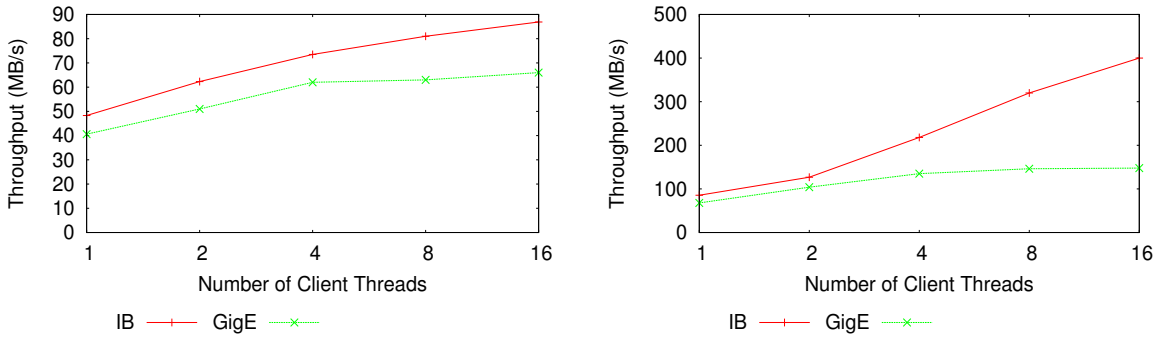


Figure 1: The three configurations used in the experiments



(a) Write Performance

(b) Read Performance

Figure 2: Performance Comparison of pNFS/PVFS2 on InfiniBand and GigE

file size be 256MB.

5.1 pNFS/PVFS2 Performance with Different Transports

This test evaluates pNFS/PVFS2 read/write throughput with InfiniBand and GigE as underlying transports respectively, to find out the advantages of using InfiniBand. We use 4 storage nodes as I/O servers, and up to 32 compute nodes as client nodes. On InfiniBand, both native IB protocol and IPoIB are used. Figure 4(a) and Figure 4(b) show that both native IB protocol and IPoIB improve throughput compared with GigE. pNFS/PVFS2 with native IB delivers write performance up to 190% better than that on GigE, and read performance up to 4 times better than on GigE.

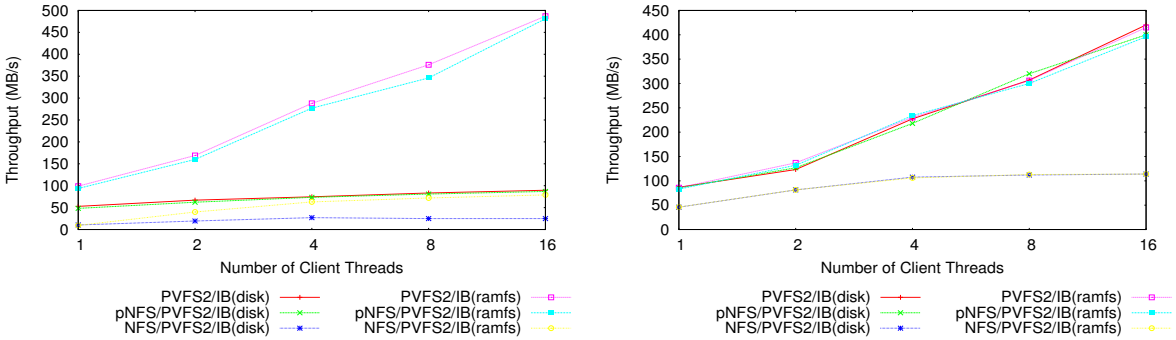
5.2 Comparison of pNFS, PVFS2 and NFS on InfiniBand

In this test we compare the read/write performance of PVFS2, pNFS/PVFS2 and NFS/PVFS2 on InfiniBand by varying the number of IOzone threads, which is also the number of client nodes. Figure 5(a) and Figure 5(b) give the write and read performance of these 3 file system configurations with 4 I/O servers. pNFS/PVFS2 yields write throughput as 490MB/s and read throughput as 2250MB/s. It is 11 and 24 times higher than that of NFS/PVFS2 for write and read respectively. Although NFS is also configureed with PVFS2 as the backend file systems, the single server bottleneck prevents it to deliver high throughput. At the same time pNFS/PVFS2 produces a a performance very close to that of PVFS2. This indicates that pNFS/PVFS2 enforces very little overhead on top of the backend PVFS2 file system.

5.3 pNFS/PVFS2 Scalability with Increasing Number of I/O Servers

In this experiment we vary the number of I/O servers to see the scalability of pNFS/PVFS2 with respective to the number of I/O servers. Figure 6(a) and Figure 6(b) dipict the read/write throughput of pNFS/PVFS2 at 4 I/O servers and 8 I/O servers with InfiniBand. From these two figures we can see that pNFS/PVFS2 with InfiniBand scales up well with respective of I/O servers. At 32 clients, pNFS with 8 I/O servers achieves write and read throughput about 754MB/s and 3100MB/s respectively. It improves performance by 77% and 50% for write and read respectively, compared with 4 I/O servers.

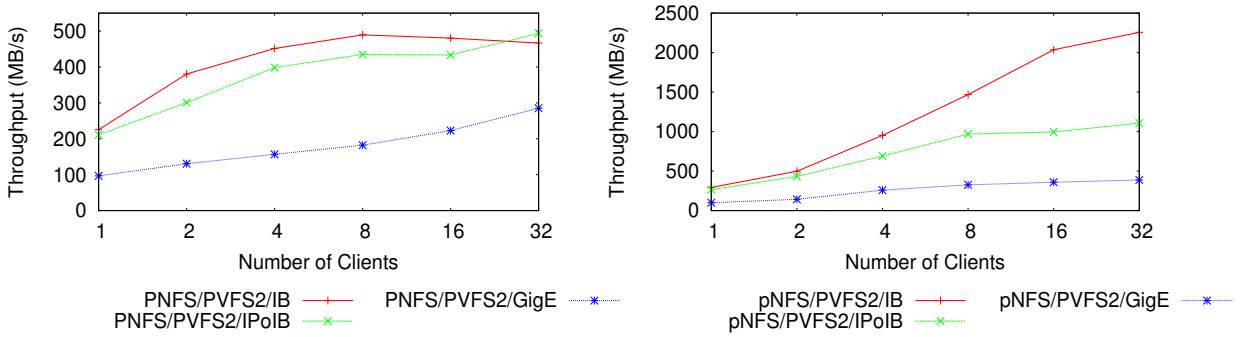
One thing to be noted is that the absolute numbers shown in Figure 6(a) and Figure 6(b) are not as high as those shown in



(a) Write Performance

(b) Read Performance

Figure 3: pNFS, PVFS2, and NFS Comparison



(a) Write Performance

(b) Read Performance

Figure 4: pNFS/PVFS2 Performance over Different Transports

previous sections. This is due to some system changes when we did this set of experiments. Since this set of numbers (with 4 and 8 I/O servers) has been taken at the same time period, the absolute lower numbers do not affect the fairness of this comparison.

6. RELATED WORK

In related work, Honeyman, et.al [6] proposed the first design and evaluation of pNFS with the PVFS2 file layout driver. Evaluation was with Gigabit Ethernet. We have evaluated pNFS with InfiniBand native transport, IPoIB and Gigabit Ethernet. In later work, Honeyman, et.al. [8, 7] also proposed different techniques for improving small read and write performance and direct-pnfs. Gigabit Ethernet was used for this study also. Native performance of PVFS2 over InfiniBand, compared to TCP is discussed by Wu, Pete and Panda [13]. In this paper we have evaluated pNFS over PVFS2 performance. With respect to improving NFS performance, we have done work in the area of NFS over RDMA in OpenSolaris [11, 10].

7. CONCLUSIONS AND FUTURE WORK

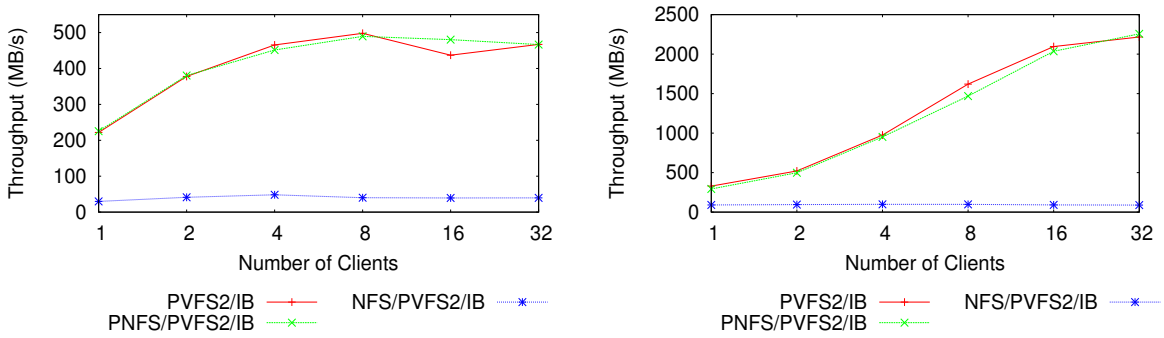
In this paper we did performance evaluation of pNFS/PVFS2 on InfiniBand clusters. From our experimental results, we are able to answer the following questions:

- What are the advantages of using InfiniBand over Gigabit Ethernet in a parallel file system environment?

  Answer: InfiniBand, both native IB protocol and IPoIB, improves pNFS performance significantly. The native IB can improve write throughput by up to 190% and read throughput by up to 480%.

- How much is the performance gain if we use pNFS instead of the traditional single server NFS?

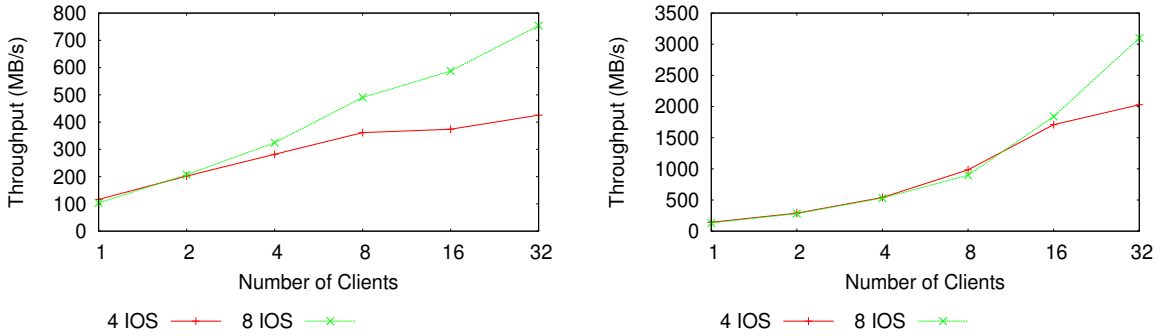
  Answer: Compared with the traditional NFS, pNFS/PVFS2 provides significantly higher throughput and shows better scalability. pNFS/PVFS2 achieves peak write throughput about 490MB/s, and read throughput about 2250MB/s, with 4 I/O servers, which is 11 and 24 times improvement over NFS respectively.



(a) Write Performance

(b) Read Performance

Figure 5: Performance Comparison of PVFS2, pNFS, and NFS on InfiniBand



(a) Write Performance

(b) Read Performance

Figure 6: pNFS/PVFS2 Performance with Different Number of I/O Servers

- Is there overhead introduced by the pNFS PVFS2 layout driver compared to a native PVFS2 installation?

  Answer: pNFS adds very little overhead and achieves almost the same throughput as the native PVFS2.

- How does pNFS scale with increasing number of I/O servers?

  Answer: pNFS scales well with increasing number of I/O servers. pNFS with 8 I/O servers achieves write and read throughput about 754MB/s and 3100MB/s respectively, at 32 clients. It improves performance by 77% and 50% for write and read respectively, compared with 4 I/O servers.

To conclude, we believe that pNFS on InfiniBand clusters is promising.

As part of future work, we would like to explore the impact on performance and scalability of pNFS with a file layout driver, using NFS/RDMA. We will also expand our experiments on larger scale clusters with application level workloads. We also want to use 10GigE/iWARP as the transport and evaluate pNFS performance.

8. ACKNOWLEDGEMENTS

This research is supported in part by DOE's Grants#DE-FC02-06ER25749 and #DE-FC02-06ER25755; NSF's Grants #CNS-0403342 and #CPA-0702675; grants from Intel, Mellanox, Cisco systems, Linux Networx and Sun Microsystems; and equipment donations from Intel, Mellanox, AMD, Apple, Appro, Dell, Microway, PathScale, IBM, SilverStorm and Sun Microsystems.

9. REFERENCES

[1] InfiniBand Trade Association. http://www.infinibandta.com.
[2] Iozone Filesystem Benchmark. http://www.iozone.org.
[3] Lustre a network clustering FS. http://wiki.lustre.org/index.php?title=Main_Page .
[4] Parallel Virtual File System. http://www.pvfs.org .
[5] CITI. Linux pNFS Kernel Development. http://www.citi.umich.edu/projects/asci/pnfs/linux .

[6] Dean Hildebrand and Peter Honeyman. Exporting storage systems in a scalable manner with pnfs. In *Proceedings of the 22nd IEEE - 13th NASA Goddard (MSST2005) Conference on Mass Storage Systems and Technologies*, April 2005.

[7] Dean Hildebrand and Peter Honeyman. Direct-pnfs: scalable, transparent, and versatile access to parallel file systems. In *HPDC*, pages 199–208, 2007.

[8] Dean Hildebrand, Lee Ward, and Peter Honeyman. Large files, small writes, and pnfs. In *ICS*, pages 116–124, 2006.

[9] IETF. NFS V4.1 Specification. http://tools.ietf.org/wg/nfsv4/draft-ietf-nfsv4-minorversion1.

[10] Ranjit Noronha, Lei Chai, Spencer Shepler, and Dhabaleswar K. Panda. Enhancing the performance of nfsv4 with rdma. In *International Workshop on Storage Network Architecture and Parallel I/Os (SNAPI)*, 2007.

[11] Ranjit Noronha, Lei Chai, Thomas Talpey, and Dhabaleswar K. Panda. Designing nfs with rdma for security, performance and scalability. In *Proceedings of International Conference on Parallel Processing (ICPP)*, 2007.

[12] OpenSolaris. OpenSolaris Project: NFS version 4.1 pNFS. http://opensolaris.org/os/project/nfsv41 .

[13] Jiesheng Wu, Pete Wyckoff, and Dhabaleswar K. Panda. PVFS over InfiniBand: Design and Performance Evaluation. In *Proceedings of the International Conference on Parallel Processing (ICPP'03)* , pages 125–132, 2003.