HPCforum
cluster computing news, opinion, technical and product information

Technical Articles

# File Systems for HPC Clusters, Part One: Distributed File Systems

**Jeffrey Layton, Ph.D.**

Posted: Sept 7, 2007

## Introduction

Clusters have become the dominant type of HPC system but that doesn't mean they aren't perfect (sounds like a Dr. Phil ad doesn't it?). While you get a huge bang for the buck from them, somehow you have to get the data to and from the processors. Moreover, some applications have fairly bengin IO requirements and others need really large amounts of IO. For either type of IO requirements you will need some type of file system for your cluster.

This two-part article is really a survey article in that I want to just touch upon file systems that are available for clusters that scratch the proverbial IO itch. My goal is to at least give you some ideas of file systems and some links so you can investigate them further. But before I start, I want to discuss some of the enabling technologies for high performance parallel file systems.

## Enabling Technologies for Cluster File Systems

As I discussed in a **previous article** Infiniband has very good performance and the price has been steadily dropping. DDR (Double Data Rate) Infiniband is now pretty much the standard for IB systems. DDR IB has a theoretical bandwidth of 20 Gbps (Giga-bits per second), a latency less than 3 microseconds (depending upon how you measure it), an N/2 of about **110 bytes** and a very high message rate. Also it's price has dropped to about $1,200-$1,400 per port (average costs). At the same time, most applications don't need all of that bandwidth. Even with multiple cores using the same IB connection for communication, there is bandwidth left over.

So we have lots of bandwidth available that applications don't really use even for multi-core systems. To make use of all the capabilities of their interconnects, vendors and customers are using that left over bandwidth for parallel file systems.

## File System Introduction and Taxonomy

To make things a little bit easier I have broken file systems for clusters into two groups: Distributed File Systems, and Parallel File Systems. The difference is that parallel file systems are exactly that, parallel. While distributed file systems are not necessarily parallel but the client and storage are separated and use a network for data transfers.

Due to the length of this article, I can only touch on a few file systems. So please don't be alarmed if there is a file system that you like and use, that is not in this article. It's not intended as an insult toward that file system. Rather, it was just a choice I made to cut the number of file systems I discuss. My apologies for the choices I have made. If you would like a specific file system discussed, please send me some email and let me know what you would like discussed and I will do my best.

## Distributed File Systems

The first set of file systems I want to discuss are what I call distributed file systems. These are file systems that are network based (i.e. the actual storage hardware is not necessarily on the nodes) but it not necessarily parallel (i.e. there may not be multiple servers that are accessing the file system). I think you will recognize some of the names of distributed file systems.

## NFS and NAS

**NFS** has been probably the primary file system for clusters and a great deal of HPC for some time. It was the first widespread file system that allowed distributed systems to share data effectively. Consequently it can be viewed as one of the enabling technologies for clusters and HPC to thrive. More over, it is the only file system **standard** for sharing data over a network.

**NFSv4** was released with some improvements. In particular it added some speed improvements, strong security was added with the ability for multiple security implementations, and even better, NFS became a stateful protocol (at least for the most part).

There are a very large number of companies that make, market, and support NAS (Network Attached Storage) devices. For example,

- **IBM**
- **Netapp**
- **EMC**
- **HP**
- **ONStor**
- **Scalable**
- **BlueArc**

and other vendors including many small hardware shops, make, market, and support NAS devices. Since the NFS protocol is a standard these devices can all interoperate. The large number of NAS vendors illustrate the popularity of NAS devices.

However NAS devices are not without problems. These problems are becoming more apparent as systems grow larger and larger. For example they don't scale well, particularly for large systems, either in terms of capacity or performance. Also, they have limited performance. To overcome some of these problems, vendors have made specialized hardware to improve scalability and performance. But there is only so much you can do improve scalability and performance within the limitations of NFS

In general a NAS device has a single NFS server that is connected to a switch that all of the cluster clients can "see." Behind the single NFS server, sometimes called a filer head, is hardware storage. The filer head exports the file system that the clients mount. When a client wants to access the data, the request is sent to the filer head that then sends back the data. The performance limitations are primarily due to all data requests having to flow through a single point. Some vendors have come up with solutions in the filer head to improve performance. Plus they have used some fairly hefty hardware to allow the storage to scale to some fixed amount (usually in the Terabyte range). There is also a class of solutions called Clustered NAS (more on that topic further down) that improve performance but can also have some scalability and performance limitations.

But the interesting thing is that many codes don't require lots of IO for good performance. These codes will run very well using NFS as a the storage protocol even for large runs (100+ nodes or several hundred cores). This is true until the input and output files for these codes become extremely large or if the code is run across a very large number of processors (in the thousands).

As I mentioned previously, NFSv4 added the ability for NFS to become a stateful protocol. NFSv4 hasn't been a big success in that not many sites are using it, but now NFS has all of the components of an industrial strength file system but it still lacks performance and scalability. But that is about to change.

## pNFS

Currently a number of vendors are working on version 4.1 of the NFS standard. One of the biggest additions to **NFSv4.1** is called **pNFS** or Parallel NFS. You might think this is an attempt to kludge NFS to have better performance and scalability, but this isn't the case. It is a well planned, tested, and executed approach to adding a true parallel file system capability to the NFS protocol. The goal is to improve performance and scalability while making the changes a standard (recall that NFS is the only true file system standard). More over this standard is designed to be used with file based, block based, and object based storage devices with an eye towards freeing customers from vendor lock-in. The NFSv4.1 draft standard contains a draft specification for pNFS that is being developed and demonstrated now. A number of vendors are working together to develop pNFS:

- **Panasas**
- **Netapp**
- **IBM**
- **Sun**

to name but a few. With vendors of this magnitude working to develop pNFS you can see that there are some heavy weight support behind it.

The basic architecture of pNFS is shown below in Figure 1.

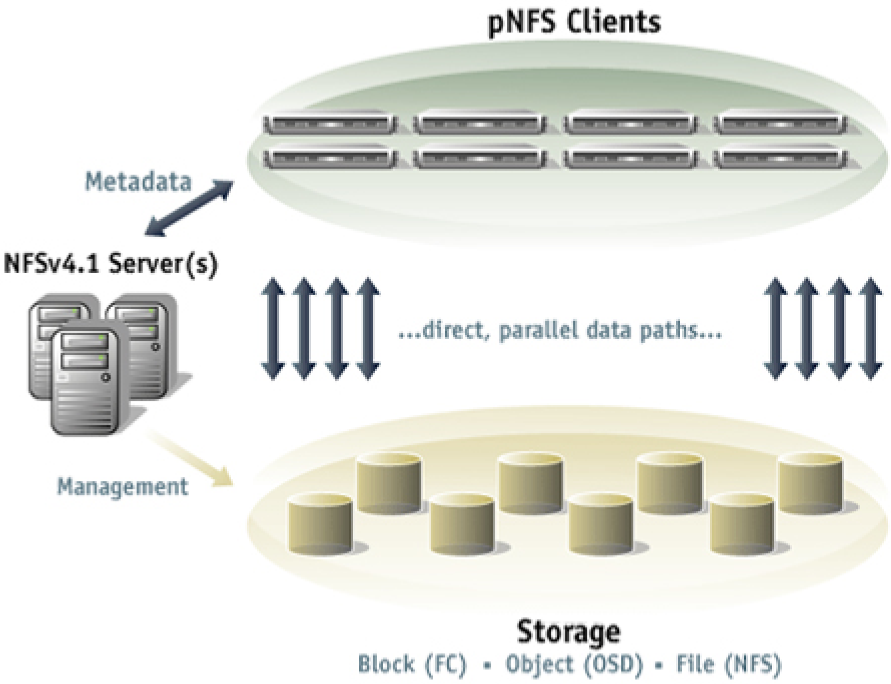
pNFS Clients
Metadata
NFSv4.1 Server(s)
...direct, parallel data paths...
Management
Storage
Block (FC) • Object (OSD) • File (NFS)

**Figure 1 - Architecture of pNFS (taken from www.pnfs.com)**

The configuration consists of some NFSv4.1 servers (shown on the left) that act as metadata servers. On the bottom of the figure is the storage that is connected to both the metadata servers and to the pNFS clients. The clients are also connected to the NFSv4.1 servers. When a client requests data, the metadata managers are contacted. They communicate with the storage to determine where the data is located (in the case of read) or where the data could be located (in the case of write). Then the metadata managers pass back a layout of where the data is located on the storage devices. The client(s) then contact the storage devices directly. This eliminates one of the biggest performance bottleneck - the metadata manager. Plus the clients can access the storage devices in parallel to improve throughput. The vendors will only need to write what is called a layout driver for the pNFS client. This allows the client to communicate with the storage pool.

One of the features of NFSv4.1 is that it avoids vendor lock-in. Part of this is because pNFS will be a standard. In fact it will be the **only** parallel file system standard. The standard allows for pNFS to use the three major types of storage, allowing many, many vendors to ship storage that can be part of a pNFS file system. This allows you, the customer, to chose whatever storage you want as long as there are layout drivers for it.

So why should vendors support NFSv4.1? While it's not totally obvious since it looks like vendors have competing products, but the answer is an important one. With pNFS, the vendors can now support multiple Operating Systems without having to port their entire software stack to the new OS. They only have to write a driver for their hardware. Writing a driver is much easier than porting an entire software stack to a new OS. So for the vendor pNFS increases the possible customer base for their storage products.

pNFS is on it's way to becoming a standard. While one can't predict when it will happen, it appears that by the end of 2007 or the beginning of 2008. If you want to learn more about pNFS you can go to this pNFS **website**. You can also go to the **Panasas** website and follow links to presentation and information about pNFS. If you want to experiment with pNFS now, the **Center for Information Technology Integration** (CITI) has some Linux 2.6 kernel patches that use PVFS2 for storage.

## Clustered NAS

Since NAS boxes only have a single server (single filer head), **Clustered NAS** systems were developed to make NAS systems more scalable and to give them more performance. A Clustered NAS uses several filer heads instead of a single one. The filer heads are then connected to storage via a network of some type or they storage may be directly attached to each filer head.

There are two primary architectures for Clustered NAS systems. In the first architecture, there are several file heads that have some storage assigned to them. The other filer heads cannot access the data, but all of the filer heads know which filer head has which data. When a data request from a client comes into a filer head, it determines where the data is located. Then it contacts the filer head that owns the data using a private storage network. The filer head that owns the data retrieves the data and sends it over the private storage network to the originating filer head which then sends the data to the client. This first approach is used by NetApp (NetApp-GX).

In the second approach the filer heads are really gateways from the clients to a parallel file system. For these types of systems, there are filer heads that communicate with the client using NFS over the client network but access the parallel file system on a private storage network. The gateways may or may have storage attached to them depending upon the specifics of the solution. This approach allows the ClusterNAS to be scaled quite large because you just add more gateways which also increases aggregate performance because there are more NFS gateways. This approach is used by **Isilon**. It is also used by Panasas, IBM's GPFS, and other parallel file systems when they are running in a "NFS mode."

The problem with either approach to Clustered NAS devices is that you have limited performance to the client because you are only using NFS as the communication protocol. Most of the Cluster NAS solutions use a single GigE connection to each client so you are limited to about 90-100 MB/s at most to each client.

## Summary

I want to stop here since this is a logical break between Distributed File Systems and parallel file systems. In part 2 of this article, I will discuss some of the more popular current parallel file systems for clusters. These file systems can be deployed today. But don't underestimate the power of pNFS in the near future. Beware the standard.

*Dr. Jeff Layton has all of his degrees in Aeronautical and Astronautical Engineering with a focus on topics that require HPC. He has been working with HPC for over 20 years and has been working with clusters for over 10 years. He has written on a variety of topics including clusters, MPI, file systems, and performance tuning. He can be reached at laytonjb@gmail.com.*

*Copyright 2007, Jeffrey B. Layton. All rights reserved.*

*This article is copyrighted by Jeffrey B. Layton. Permission to use any part of the article or the entire article must be obtained in writing from Jeffrey B. Layton.*