Building and Managing High Performance, Scalable, Commodity Mass Storage Systems

John Lekashman
NAS Systems Division
NASA Ames Research Center
Mail Stop 258-5
Moffett Field, CA, 94035
lekash@nas.nasa.gov
Tel: +1 650-604-4359
Fax: +1 650-604-4377

Abstract

The NAS Systems Division has recently embarked on a significant new way of handling the mass storage problem. One of the basic goals of this new development are to build systems at very large capacity and high performance, yet have the advantages of commodity products. The central design philosophy is to build storage systems the way the Internet was built. Competitive, survivable, expandable, and wide open.

The thrust of this paper is to describe the motivation for this effort, what we mean by commodity mass storage, what the implications are for a facility that performs such an action, and where we think it will lead.

We believe the implications for the future are that it will become much easier to use storage, resulting in extreme growth in deployed massive storage capacity,

1. Introduction

This paper is about managing commodity, high performance, long term storage.

Commodities are things that you can get from a lot of different places.

Management is something you can hardly get anywhere.

High Performance is an elusive target. In cars, it often inspires a sense of pleasure to be driving a high performance vehicle. In computing systems, it, usually meaning that the thing in question is performing its intended function well enough that something else gets on your nerves.

2. Our work

We feel our work is very useful, because we have figured out how to put all three of these together.

We're the first ones to build such a system, where the proprietary nature of the system is limited to a given system unit, such that it no longer will annoy us.

2.1. Our data is ours.

No one can hold it hostage. The key point of building data storage systems, after all, is to store your data. Since we are the ones doing the data storage, it only makes sense that its our data, and not subject to random external constraints, licensing arrangements, or other encumbrances that make facility management irritating.

This means that all the hardware and software interfaces have to be wide open. Things like TCP/IP, RAID, SCSI, Fibre Channel, OpenVault, DMAPI, and so on, are key, and must be the total definition of the system.

No vendor proprietary file systems, no charging per byte stored, no vendor specific implementation can exist in the system across an interface boundary. And, the whole part that any vendor provides must conform to the interfaces that are defined to the other parts.

So, it is perfectly ok for the vendor of a host platform that accesses your data to have their own native, proprietary file system. Its not ok for your data to be stored in that file system in such a way that it takes more than a day or so to read all the data out of that file system. More than a day or so will annoy you, and then its not high performance.

2.2. Scale matters, both up and down.

Table 1 shows some current choices for the various parts of the hardware system.

Figure 1 shows how to connect these parts, in a relatively generic fashion.

Its important to note that there is wide dynamic range in what can be done. At the low end, a storage system with a capacity of 1/2 a terabyte can be built for less than $30K, using PCs, ethernet, and 4mm tape changers. At the high end, a storage system could be built for $15M with a capacity of 3 Petabytes. The hardware cost of the low end system is about 2 cents/MB, the high end, .5 cents/MB. One can continue to scale up the size, however, there are few cost reductions. This kind of cost pattern, where the size has a range over five orders of magnitude, the overall cost per unit drops some, due to large purchasing power, shows the hardware involved in these systems is truly commodity based.

This then, is the hardware commodity base. The software, however, in current large scale storage systems, is anything but a commodity. There are a number of different vendors, each pursuing their narrow customer base. Their data formats are incompatible, their file systems proprietary, or only run on one kind of very specific hardware. When one discusses transitions with operations managers, they talk in terms of years, and hundreds of thousands of dollars to effect the transition. This then, is nothing like a commodity, and really, nothing like high performance. Remember, high performance things are not annoying. Years of transition, and much cost, are.

In order to win, the software needs to play the same tune as the hardware components. There are several key ideas to follow to be able to get there. The environment is now correct for this to occur, with a few key steps.

Interconnect Fabric	RAID Storage	Deep Archive	Archive Server	File, Program, Data Servers
HIPPI	Megadrive	STK	Pentium PC	DEC Alpha 8200
Fibre Channel		IBM		SGI Origin 2000
Fast Ethernet	Kingston	4mm SCSI	Sun desktop	Pentium PC
...	...	...	...	...

Table 1. Component parts.

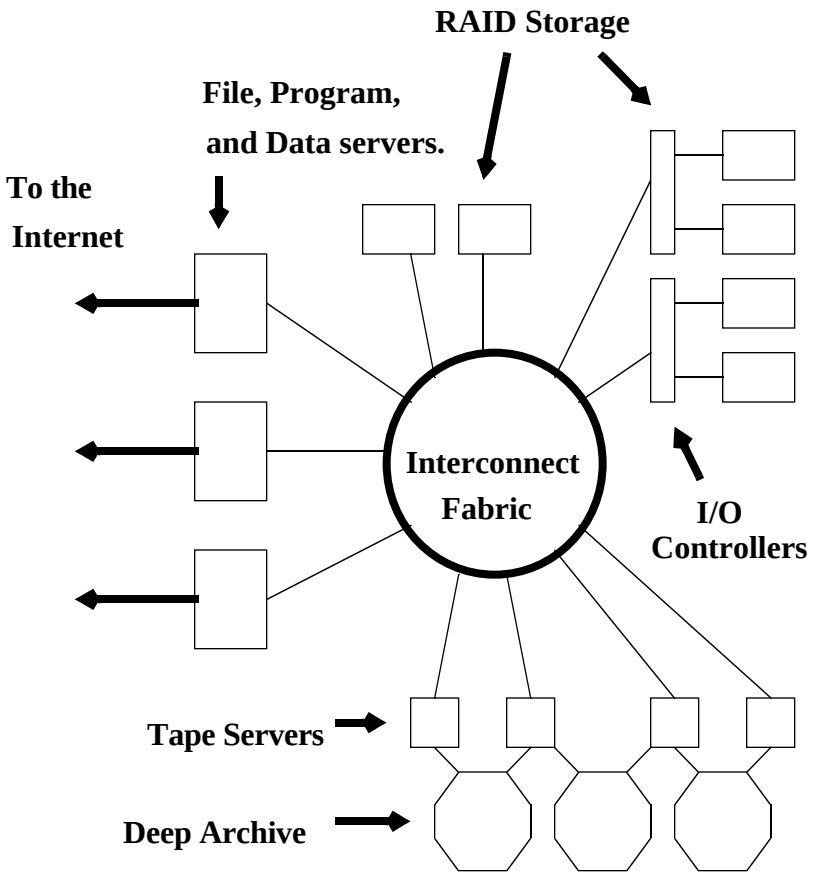

Figure 1: The hardware components of a mass storage system.
After this, its a mere matter of software.

2.3. Key enabling mindset.

There are two key ideas that are needed to be able to do this. The first is to adopt a network point of view. The second is to recognize what that means to the software.

2.3.1. You must have a network centric point of view.

The basis model for the Internet is that components are designed to be able to plug into each other, virtually without limit. As shown in Figure 1, the various hardware components are designed to do this, and can easily be so connected. However, the business model of the site management needs to understand this as well. It is a common thought process to think of 'my storage machine's disk drives', or my storage machine's tape drives'. One must lose this mindset, and think of peripherals that happen to be hanging around the network. They are a common resource, all of them. No machine owns anything that has a long lifetime. Only by doing this can you avoid the deadly, time consuming and costly problem of large system transition.

2.3.2. Its the API, stupid.

Here we steal a phrase from our president some years ago, and mangle it slightly. This software interface drives what it is you can do. One can make all the comments one likes about simply plugging hardware components together, but if the software doesn't play together well, nothing actually works. So, once again, the overall storage system mindset has to go, and the Internet survivable and interoperable mindset needs to come into play.

Here is the key idea: You need to have your software laid out such that you are prepared to throw everything away, at a moments notice, within a given scope.

To reinforce it, I will say it again, slightly differently: In order to save your data, you must be prepared to abandon any part of it, at any time.

That scope can vary widely, depending on the business model of your facility. A small company, for example, might agonize over a $3000 tape drive. A large national facility may be prepared to roll a $300K network fabric out the door, because a vendor is no longer responsive. This kind of scoping is an essential part of the business management that needs to take place, in order to effective build the facility. Exactly where your facility exists financially is an essential part of the scoping effort.

3. How do we get there?

There are a host of implications to conducting such a development.

Open software interfaces have to be defined, where they do not exist. This process has in fact been going on for years, with the development of RAID, the definition of DMIG and DMAPI, the creation of the Openvault tape specification, IPI channel interfaces, SCSI and Fibre Channel disk interconnect, and the whole Internet standardization process. The time is now right to build a mass storage system that takes advantage of all of them.

4. Related and Future Work

There has been much related work in the IEEE Mass Storage Reference Model, and in particular in the development of HPSS (High Performance Storage System) [1]. Many of the ideas being developed are similar, such as network centric storage, and well defined APIs to store data. The key factor in this work is that the storage system software becomes completely unencumbered.

There is software glue to be created to be prove this concept. The NAS facility has been putting together a reference implementation for the past year, showing how it can be done.[2]

5. Conclusions

Really big storage systems are going to cost a lot less. Products will have to compete on capacity pricing with the cost of an IDE disk drive down at the local computer store. For disk, this is on the order of 10 - cents a megabyte, in mid 1997. Tape is of course 2 - 3 orders of magnitude cheaper.

Customers will be less nervous about deploying such systems, because they don't have the long term capital outlay and investment.

We believe that the most significant result of this type of work will be a great deal of growth in the marketplace.

Our basic design philosophy is mirroring that which we put into building the Internet, many years ago. Big storage systems will eventually be as easy to put together as that has now become, and because of that, people will do it. There is no question that this will take many years to get such ubiquity to occur, and the end results will certainly look nothing like what we first build. But it will happen.

References

In keeping with the network centric view, the major references used in this work are on the net. There are, no doubt many related papers as well. They are also shown at these locations. However, on the net is where the action is taking place., and where information for this paper was created.

[1] http://www.sdsc.edu/hpss/ - The web home for the High Performance Storage System.

[2] http://science.nas.nasa.gov/Groups/NAStore - The web home for storage at the NAS facility.