Approaching A Petabyte

A Case Study, and Random Thoughts

Hal Miller

Department of Molecular Biotechnology
University of Washington
Seattle, Washington

October 18, 1999

ABSTRACT

We know how to grow computing sites incrementally, aiming into the terabytes of storage. This paper looks at some issues involved in massive growth of online storage needs. It presents a series of questions that the system administrator needs to contemplate before designing significant expansion or a very large new site, and provides some commentary on one recent case study.

1 INTRODUCTION

I remember my first machine. 4K RAM, cassette tape drive, Intel 4004. When we moved to our first Z80, we expanded to 16K RAM, and an 80K (count 'em, 80!) floppy. We had all the elbow room we imagined that one could possibly use, and thought we'd never fill it all at once.

This year I turned on Phase 1 of a new project site: 50 CPUs, 40GB RAM, 8 terabytes of disk, 22 tape drives, etc. The plan calls for, in three more phases, reaching a petabyte of main storage in another couple of years.

What is a petabyte? The obvious and simple answer is "1024 terabytes". That doesn't really have much meaning to most people, so let's look at what's behind the term. Suppressing the urge to walk through personal history from that 80K floppy drive to the present, we note that only a year or so ago, you had to order the special options to get a 1 gig disk on your new PC or Macintosh. Suns and SGIs came with 1 gb internal disks. HPs contained a 2gb drive each. A year ago, big servers contained a couple hundred gigabytes. "Really big" sites contained a few terabytes. Backups consisted of a period of offline time for a "full" dump, or some additional level of risk while doing a "hot" run, plus nightly "incremental" efforts. A year ago, we had plenty of difficulty keeping up with our system and network architecture issues as we grew to a half terabyte. Only one year ago, nobody thought of a petabyte, even in jest. We mirror all the disk space on some sites, we keep multiple copies of full dump tapes, we run little jobs that look for files that might no longer be needed (e.g. "core"), we run HSM (Hierarchical Storage Management). We figure that our half terabyte should last a while if we do these things. But, what happens when your users come to you (as one of mine recently did) and ask, "what comes after petabyte?" (The answer, in 1024-incremental naming, is 'exabyte', 'zettabyte' and 'yottabyte'. After that nobody has yet seen fit to name - stand by....) In short, we don't really know what a petabyte is, because

(other than perhaps one or two) there aren't any sites with enough storage to find out.

So, what does it mean? Is it really an issue? If so, what are the problems we are about to face, and how do we go about solving them before they overwhelm us at the last moment (as most such problems tend to do)?

The first issue to deal with is whether and why these questions are relevant. I ran numbers on the storage requirements for our new site as I worked with the researchers in scoping out the computing environment. I extrapolated from current space used and current number of analysis runs per day, coupled with the user-predicted average number of daily runs under new grants. I don't know whether he will ever actually need it, but the user noted above rewrote his software to handle 9.5 exabyte file sizes. That's per file. This is a genetic sequencing/genomics site, one of many such around the world. Other "industries" are also consuming massive quantities of storage space. Partly due to the low cost of disk, partly because of the ever-increasing "piggishness" of software packages, partly because of what arenas computers have expanded into, and partly based on the rule that disk usage will expand to consume all available disk space, the need for very large storage installations is real, and we had better begin facing it.

This paper is a survey of some of the issues involved in expanding to "very large" storage quantities.

1.1 Trends in computing

Article after article claims, study after study shows, and more organizations are faced daily with the fact that computing requirements are growing at an unprecedented rate. The "weakest link" in the computing "chain" is, unfortunately, the place where most of the growth is occurring: storage and the use of I/O bandwidth. While users constantly talk about faster processors and more RAM, most currently available software really doesn't need

nor use increases in those areas. What *is* required is more disk space.

The computing process used to be viewed as a flow of data through a set of constant procedures, that data being "acted upon" along the way, and producing some final version as an output. This is shifting to a data-centric view, where the data is the core "constant", and the processes flow and act upon/with that data. This changed trend becomes particularly important in multiprocessor, parallel, and clustering arrangements.

Despite significant marketing dollars to the contrary, the day of the "desktop" machine, PC or other, is coming to a close. People are already beginning to rely on handhelds (Pilot, HP, WinCE, etc.) for a computing interface. This trend is likely to continue - not only continue but to grow explosively. Your GPS-connected automobile will not carry all the storage it will need, nor will your Dick Tracy wristwatch. Another round of "glass house" server rooms is coming into play, perhaps with a different set of rules than that of the old days. This time, the glass house will supply a "storage dialtone", meaning users will see (read "demand") storage as an unlimited resource, always available to them. People will continue to want access to far more information than that with which they can possible make use, but the unwieldy interface of a desktop will fade into the sunset.

This migration back to glass houses is not limited to the storage dialtone, but includes a more generic classification of "computing dialtone". As users get further from the "user-to-computer" interface model, they will expect computing service when, how, and as required, in a fully ubiquitous manner. People will no longer think of themselves as "computer users" as they do whatever it is their business requires, and the computing will be done as a background service. The model of electricity or telephony will apply.

Our task is to look deeply enough into the future to be able to start shaping and creating those glass houses now. The cost of developing and building them will be far greater than that of developing the new forms of

"interface". People will be ready for the glass house (and probably unwilling to pay for it) before we are ready to provide the service pipeline.

2 PROBLEM STATEMENT

What is the problem to be solved here? Or, more precisely, what are the *problems* to be solved? Beware that this paper presents lots of questions, and only a few answers....

2.1 Preliminary Sizing Issues

First, some eye-glazing numbers. Using, for an example, the 18gb fibre channel disk drive (which is what I installed), how many spindles are involved? To get a petabyte of space after formatting overhead, not counting RAID overhead, mirroring, replication, boot disks, etc., some simple arithmetic:

*1.8pb raw == 18gb/disk * 100,000 spindles*

Now, add mirroring. As we'll discuss, backups will take forever, so you will need to consider backing up a mirror instead of a "live" set of storage. In fact, they'll take so long, you won't be able to complete a full backup before needing to begin the next one or two incrementals at the least, so you'll need at least a 3-way or 4-way mirror. My calculations indicate a 5-way is required to provide just a slight margin of safety.

*100,000 spindles * 5-way mirroring == 500,000 spindles*

Next, offsite mirroring. In many cases the data needs to be available to users around the world despite the fact that Seattle dropped off the face of the earth in an earthquake (or similar catastrophe for your location). If I have 100,000 disks carrying a single set of data, you'll need 100,000 to cover a copy of it, as will someone else. And, I'll need 100,000 to cover a copy of yours, plus another 100,000 to copy that third party's data. Now we're up to 700,000 spindles on my site.

Another way to handle the backups, instead of mirroring, is the "snapshot" technique. It still takes additional spindles, although fewer than a full mirror. We still need to consider mirroring, particularly the offsite variety, but the total count will be lower. Of course, if I snapshot "your" data, I'll need more than 100,000 to mirror your site, partially making up the difference.

Add RAID overhead, hot spares, boot disks, etc. Reaching a million spindles isn't out of the question. Just using really rough numbers for a ballpark estimate:

*1,000,000 spindles * $1,000 per disk == $1 billion to buy the disks alone*

You also need servers, towers and shelves to put them in, power supplies, host bus adapter cards, cabling, etc.

Backups - assume we use a DLT7000, with real-world compression results, and get an average of 35gb per tape.

1.8pb / 35gb per tape == 51,000+ tapes

*51,000 tapes * $70 per tape == $3.5 million for a single backup set*

Yes, we could use 50gb disks. They're slower, with more of an impact on RAID rebuild times, etc., but it does cut down some on space problems. Yes, you might find cheaper disks/tapes/whatever. Yes, you may wish to use snapshots exclusively and not mirror at all, or at least not offsite. However, no matter how you "fudge" the numbers, no matter how much you may pick on the above details, the totals are huge compared with what most of us, and most of our managers and finance people, are used to dealing with.

2.2 Problem Classifications

I classify the problems into four categories. The first is theory based. The other three are practical: costs, performance issues, and impact (both upon users and sysadmins.)

2.2.1 Theory Problems

From a theoretical standpoint, we need to ask some big questions. The level of impact this quantity of storage has upon system design, operation, and maintenance is so significant that it breaks many of our currently accepted "truths". Some of this will become more evident as we delve into the costs, below.

When thinking about petabytes of online storage, after the glee over the tremendous "toy factor", we quickly hit the reality wall full-on with the realization that we actually have to make it work. This leads us to the following questions:

> What is data? Why do we keep it? How do we manage it? What is a backup? How should we make data available? What should it look like? Where is it?

Having posed these questions, it would be considered polite to now proffer some answers. Unfortunately, good answers don't yet exist. We will need to discuss and debate these questions within this organization, and amongst the vendors, starting now.

As a starting point, here are some simplistic thoughts, mostly (of course) in question format.

We might look at data as our inventory, the way a grocer views boxes of cereal, apples, and bottles of milk. These items aren't there for our (or the grocer's) benefit, but for that of our customers who use our systems. Data may be viewed as a commodity. Perhaps it is "intelligence", in the sense of being a business or military asset. Some of its value is stand-alone, some only in context - it makes sense (or additional sense) because it is tied with other objects, be they additional data, or processes. How does "metadata" fit into the definition?

It is clear that we as sysadmins will run into disagreements with users from time to time as to the value of their keeping some particular set of files on line, on a content- or redundancy-basis. Certainly it is our job to ensure that the security and integrity of data is maintained so users don't have to keep multiple copies. Is it our position to judge on the content side? If not us, then who? How do they do this? Quality reviews? What standards do they apply? I am certainly *not* suggesting that it should fall to sysadmins, but it is an issue that must be dealt with when building a massive storage site.

Once the content and quantity is somehow regimented, how do we protect it from unintentional loss? (Intentional damage applies here too, but is only different in the security aspects, which are mostly the same as we face today.) Is the idea of nightly backups no longer relevant? If we mirror and/or snapshot, people can get back a file they mistakenly deleted this morning. If they deleted something two years ago, how many of us have that on our backup tapes now? Had they, at the time said "I want to archive this for some rainy day in the distant future", we may have put it on a separate tape or CD, so is the answer a combination of HSM, archival and snapshots? That saves us a great deal of expense, effort and bandwidth, although at the cost of yet more disk space. Do we back up only the weekly snapshot? Nightly? In any case, there is that "acceptable window of risk" issue - someone will create a file just after the backed-up snapshot, delete it just before the next one, and want it back, all regardless of the length of time we use as a periodicity for the snapshots/backups.

One security aspect that is different here than in our current scenario is that of the division of assets. The site I built (IQSB case study below) will be used as a single central computing facility for many independent organizations, with many rules and needs for keeping their data and processing separate. Yet, at the same time, some of those organizations, in varying combinations, will be working jointly on some of that data. We need to es-

tablish (and prove out) ways to ensure that subsets of data, processor time, RAM contents, etc., are available jointly between given partners, while other subsets are not. Someone coined the phrase for me: "overlapping private universes" as opposed to a single "shared universe".

Does the storage format make a difference to users, or is it entirely within our purview so long as we present it back to them in a way they are comfortable with? If we change the "standard" format (which standard are you using now, by the way?) will we be causing backward-compatibility problems? Will we be better able to handle future developments? It seems to me that our industry has dedicated itself a little too strongly to backward compatibility, and has, because of that, limited its advance pretty severely. Eventually, after due diligence and consideration, we need to be willing to cut our "losses" and proceed on to new things, absorbing the cost of converting if required. Given the size of the impact that massive storage will make, this may be one of those times to change. If so, how?

Finding specific information in a system of this size will require some change in how we deal with things now. There are lessons to be learned from other areas of computing that have, for different reasons, faced the same problem. The answers derived in those area may or may not apply, but lessons learned should be reviewed. Web search engines are probably not the answer here. Is Prospero? AFS? Some form of indexing? How about the Dewey Decimal System from our libraries?

2.2.2 Practical Problems

Back to the more practical side, we now look at the other three classifications.

1. Costs

   It should be obvious to all that building a very large site will be

expensive. Just how expensive it will be to build and to maintain may be shocking. It may be easier to break the costs down into several areas, and try to gain understanding of each separately.

- facilities: How much space is involved? Is power available? Air conditioning? Is there sufficient physical access? Network support? What sort of floor weight loading factors are involved? Environmental and/or earthquake protection? Storage space for backup tape sets, plus blanks for the "next" set? Offsite backup storage arrangements?
- hardware maintenance: What is the MTBF for the disks? Infant mortality? Burn-in time? Where do you put spares? With this many spindles, you need a large on-site spares kit, a number of people to change out disks constantly, a high level of maintenance support contracts, plus a large credit line at every local bank in your community. Given an MTBF (Seagate 18gb disk, per the Seagate site) of 1,000,000 hours, assuming 24 hour per day operation, 7 days per week, at 100,000 spindles you would need to change a disk out (after a RAID rebuild completed) every 10 hours. At 1 million spindles, it's a change every hour. RAID rebuild times vary, depending on what level of service you wish to provide to users throughout, but in most cases, it will take well in excess of 10 hours per rebuild. Given "normal" RAID environments where losing more than one disk of a set means you'll need to recover from tape, this means either going to the expensive multi-level RAIDs or cutting back on user service level (all the time) to dedicate more resource toward rebuilds. Assuming the RAID rebuilds will finish in time (a clearly false assumption), and that you have someone standing by to immediately swap the disks, you're in great shape.

- diagnostics time: Support time increases fast as you multiply the number of problems by the complexity of the system and breadth of the field in which each problem may originate. More hardware means more places for failure, in more combinations, with more red-herring misleading symptoms, and more trouble trying to diagnose. This takes some highly-developed trouble-shooting skills to manage.
- RAID design: Most RAID categories expect a fair amount of "overhead" disk, which makes sense given its purpose, but that grows at a rate that may be prohibitive as the number of data spindles explodes.
- purchase dollar total: Notwithstanding the old joke about "losing a few cents on each, but making it up in volume", while cents per megabyte may continue to fall, the large quantity, plus controllers, etc., ends up being quite expensive.

2. Performance Issues

Adding a very large quantity of hardware to our currently "understood" configurations creates a lot of traps for the unwary. Having a few million dollars worth of computing equipment won't do anyone any good if some SCSI drives and a couple of PCs running Beowulf will outperform it. Some of the ankle-twisting gopher holes directly impact design, while others need to be addressed during implementation.

- filesystem limitations: Most operating systems have filesystem and filesize limitations that no longer serve user needs. How quickly can a filesystem locate and return a given item of data when we don't know where it may have been put?
- I/O performance: As we migrate back to a central server operation to handle exported network file service, each client platform's capability to keep up is strained. Each server is likely to

have more clients than most folks assumed would be the case in the current definition of "client-server" computing.

- NFS performance: We've been toying with this one for years, but the quantum leap in storage puts a whole new perspective on the tuning issues.
- lack of SAN (Storage Area Network) standards: Interoperability of hardware and software is (surprise, surprise) a big problem.
- SCSI limitations: The SCSI bus is limited to 7 (or 15 for Ultra) devices per instance, meaning that you need to keep adding controllers to your CPU/ server box to expand total storage space. Most servers run out of card space pretty quickly. While other buses exist (e.g., HIPPI, IDE), often in fairly significant numbers, they don't have anywhere close to the market penetration, standardization, widespread acceptance, nor in some cases the performance to replace SCSI on servers, so the limitations remain issues for us.
- vendor filesystem interoperability: Most of us run some software package or two that gives some (often limited) subset of functionality of differing filesystem types on the target machine(s). Examples are CAP, Netatalk, SAMBA, and packages that allow such as UFS, VxFS, JFS and/or XFS to co-exist. Some of these are based on user-owner permissions (UFS), some on group permissions (NTFS), and are not compatible. File locking methodologies differ significantly, particularly with regard to multiple machines having joint write access to common files.
- filesystem versus database: In the "theory" section above is a question regarding just what data really mean, and another asking about how to present it to users. In addition, how we should store it is no longer clear. Is the concept of "filesystem" as we

know it obsolete? Should we be putting *everything* into some kind of database? Clearly, given applications such as I have that hold tens of thousands of files in each of tens of thousands of sub-directories in multiple hierarchies, we have outgrown the UNIX File System as it is currently implemented on most machines.

- high availability: Current technology gives us some ability to fail-over certain services from a machine that crashes to another one preconfigured to jump in as required. A "relatively well-known technology" service that can be failed-over is NFS. Given two machines, in any of a number of configurations, after a crash the second one can serve clients as if the first were still on line. This is somewhat complex to implement, but certainly not a killer. Other services can be migrated as well, but they tend to be much more difficult. We ran into the problem, for example, of failing over an Oracle instance, where Oracle and Veritas don't exactly speak the same language. For the most part, the only service really available in HA mode is NFS, and it has some noticeable impact on the surviving server. Getting the load redistributed after reboot of the failed server is also not a simple thing. The HA configuration assumes that the two servers are sitting side by side, thus having inter-site fail-over doesn't yet seem to be a reality.
- replication and mirroring: Where data must be made "always available" to external sites as well as to your own, it often makes sense to create a second (or additional) copy somewhere else. Not only will the disk vendors be eager and willing to assist you in doing this, your real property, power and air conditioning contractors will start buying you holiday gifts. If you are using JBOD (Just a Bunch Of Disks), you will need to protect yourself against loss of a disk, which more than likely means either software RAID,

or complete site multi-way mirroring (at least tripling or quadrupling your spindle count). Buy disk manufacturer stock now. Also remember that mirroring or replication in any form will require a big chunk of network bandwidth, plus some CPU power. How do the physical and topological distances between sites affect this scenario?

- cacheing: This is a technology that nobody in our business can do without. At the same time, it can kill us. Given the quantity of storage we are discussing, the number of humans likely to be accessing files simultaneously, the size of filesystems and files, and the number of machines each thinking they own a given file simultaneously, all current cacheing paradigms break. Performance of this big of a storage system requires some very fast cache at the controller level at least, with large buffering, but the more bits cached and buffered, the more likely it is that a block will be changed under you before your flush timeout.
- disaster recovery: We should all, of course, have in place a set of disaster recovery and business resumption plans. When dealing with what we now call a "normal" site, we know how to lay out the costs, vendors, and rebuild times. We figure on alternate sites from which to work while our building is being reconstructed. We have a list of who has sufficient resources available, with agreements in place, to get our most critical users back on line while we rebuild. How do these plans look when the quantity of storage is so large?

3. Impact

Remember, someone (presumably you) has to maintain all this stuff, and, like it or not, there is a rule someplace that still limits you to some fraction of 24 hours per day available to work on it. There

is another rule that our job is "overhead", thus always underfunded. Remember that your users are going to come to the new environment with a bazillion questions, and the ever- proven ability to mis-keystroke everything. They won't like what they see because it's "different", they don't understand it, or what not. Fixing their problems by this implementation doesn't mean that they're now out of problems for you, nor that they're happy. Some specific areas to consider:

- asset control: How does one keep track of all this hardware and software?
- backup schemes: What sort of offsite tape storage do you need? What is your full dump cycle? When do you do incrementals? How long does it take to complete a backup? How much change can you afford to miss when files may be created and deleted during that cycle (what is your "acceptable window of risk")?
- ability to use: How does a user find data? What operating systems can actually access this size operation? What applications can? What do we do regarding the inevitable users who don't clean up? They cause us problems when they only have a few gigs available to soak up - how do we handle the "no quotas here" sites with a petabyte?
- failure of existing tools: Try doing a 'find' or 'df', or waiting on a 'du -sk', in a very large directory in a multi-hundred terabyte filesystem.
- interoperability across operating systems, filesystem types, vendors: Will a given client be able to access, read and understand what another client has written?
- support time, architect/configuration time: Who can handle this installation? How many sysadmins does it take to screw in all those light bulbs? To change them when they burn out?

3 WHO FACES THIS SET OF PROBLEMS?

In one sense or other, whether now or later, all of us face these issues. Until now, most of us have handled growth "at sufferance". Needs grew, we added hardware to the existing environment, and just kept it going. This model breaks as we expand it this dramatically. Some of us have already reached the point of the problem: the motion picture industry, oil companies, and biotechnical laboratories, climatology, meteorology and atmospheric science laboratories, oceanographic research groups, telcos, etc. Who may be next? Libraries, airlines, large manufacturers (aircraft, automobiles, electronics), universities, hospitals, "on-line audio" houses.

As we migrate computing away from the model of user-to-computer interface and toward a "dial-tone" provision of computing services, the growth will be in a new round of glass house server rooms. These will need to face data storage requirements unparalleled in computing history. The change in user interface devices is already occurring, and computing has become prevalent in new and massive areas such as medical imaging. In general, computing has changed from "over there" to "right here", becoming an integrated part of daily life for a very significant portion of the population. Anyone supporting any portion of this new "infrastructure" faces these problems.

4 DEFINITIONS

Just to get us onto the same page, here are some key buzzwords and acronyms.

- backup: a process that produces an image of (particularly) user data on some independently-saved media. May be looked at in three categories: local (disk and tape on same host), client (disk host is dumped

via LAN to tape host), and SAN (disk, tape and hosts connected by SAN, using "LAN-Free", and eventually "Server-Free" backup).

- FC: Fibre Channel. Serial interface, typically 1.06 gigabaud, or about 100 megabytes/second per link. Can run over copper or fibre, carries "messaging protocols" (e.g., IP, ATM) and/or "storage protocols" (e.g., SCSI, HIPPI), and operates at 1gbps to 4gbps per link. 5 layer protocol. 3 device types: direct-connect (point-to-point), Arbitrated Loop (hubs, 126 devices), or switched (scalable to very large). In a switched "fabric", devices "log on" to the fabric as equal players.

- FC-AL: Fibre Channel-Arbitrated Loop. Rough equivalent to a LAN in fibre channel networks. Allows 126 devices on an equivalent to a broadcast domain "bus". Some vendors supply "switched FC-AL", providing in effect a switched fabric of loops instead of devices, further scaling the number of devices available.

- GBIC: GigaBit Interface Connector. Pluggable choice of copper or fibre into installed equipment. Allows hot swap of cable type on running machines and disk. Serial, media-independent interface.

- GLM: Gigabit Link-or-Loop Module. Similar to GBIC, but parallel.

- HA: High Availability. Formerly meant "available 99.999% of the time", but now more commonly, "100% of the time", or "always available". Typically implemented in multiple hosts with fail-over capability, on a service-by-service basis.

- LAN-Free Backup: Moving block data between multiple servers and storage devices instead of files over IP on enterprise network.

- NAS: Network Attached Storage. Integrated storage system attached to a messaging network, e.g. IP based ethernet. Acts as a server for

storage, typically via NFS, has a processor, operating system of some type, processes file I/O commands. See "SAN".

- RAID: Redundant Array of Independent Disks. A series of (usually) SCSI disks behind a controller, where the set appear to the the operating system as a single logical disk drive. Various "levels" exist, covering performance, reliability and recovery from disk error issues. See Appendix 1 for a description of those levels.

- SAN: Storage Area Network. Separate computer network, typically based on a "fabric" of fibre channel, switches and hubs, that connects storage devices to a heterogeneous set of servers on a many-to-many basis. Can also enable direct storage-to-storage connectivity. Processes block I/O commands. "SCSI on steroids."

- SCSI: Small Computer [Standard or System] Interface. ANSI standard that defines an I/O bus and logical interfaces supporting that bus, for the interconnecting of computers and peripherals. Primarily used to attach storage devices to CPU units.

- Server-Free Backup: LAN-Free implementation between storage devices, using a "data mover" instead of directly involved server.

- Storage Routing: As with "normal" networking with "messaging protocols" (IP stack), allows sharing of storage peripherals beyond direct host connectivity by routing SCSI and other "storage protocols". May encapsulate storage protocols over a messaging system or vice versa.

- Storage Router: Device on SAN that handles storage protocol routing, and typically also acts as a "data mover" to initiate storage device to storage device transfers (e.g. backups).

- Zoning: create VLAN- or VPN-equivalent on switch ports. Can be cascaded across multiple switches.

5 SURVEY OF TECHNOLOGY

In the absence of a complete technical solution to our problem set, there are many technology-based solutions to parts of our problems. This is a quick survey of some of the approaches we might take, and what is currently out there to fill those parts, along with some of the issues involved with these technologies.

5.1 Potential Solution Technologies

5.1.1 Filesystems

With regard to filesystems and data service, there are a number of approaches. The most common current filesystem types, each with advantages and disadvantages, include such as the UNIX File System (UFS), Network File System (NFS), Andrew File System (AFS, aka DFS), NT File System (NTFS), DOS's FAT, SGI's XFS, and the various vendors' logical volume managers (herein referred to as LVM). None of them, alone and as they currently exist, can handle the problems we are already seeing, let alone foreseeing. One though, the LVM, rates some special mention here.

The typical storage server has a "pile" of directly-connected (JBOD SCSI) disk drives, each with one or more filesystems mounted onto the server. One or more of those filesystems are "exported" to the various clients. Each operating system has a limitation on what size those filesystems may be. Most have strict limitations requiring filesystems to be mapped directly to one disk only, although allow for a disk to contain multiple filesystems as sub-parts. Thus, the largest filesystem size is controlled by a few factors: size of the disk, limitations of the operating system, limitations of the tools available, and limitations of the application software needing to use it. Hearkening back to the example above of a user writing code to handle multiple-exabyte files, we see that this model is breaking down. Various vendors (e.g.,

HP, Sun, IBM, SGI, Veritas) have come up with software packages that, often with kernel modifications, present the operating system with views it can understand, yet maintain multi-disk filesystems. These packages are generally known as Logical Volume Managers. The concept of "filesystem" is generalized to "volume", and may contain parts of "filesystems", parts or all of various disks, and in some cases, storage devices from different machines. These packages allow for growing, shrinking, or even relocating a volume, while users are using it. They also often combine with software RAID, or other technology valuable for other reasons. In most cases they will work just fine with hardware RAID or other equipment, as they don't look at the hardware as "disks" but as "logical storage devices", which may consist of a single disk, multiple disks, or units which contain multiple iterations of some storage technology. The cost is generally two to three percent of the CPU on the server, nothing on the client. Older clients often have problems with, or simply do not understand at all the large "partitions".

The filesystems we know are hierarchical trees. On one machine with a few disks, this is easy to handle. If your user put something somewhere, you'll have very little difficulty finding it. What about the case where your user can't recall where a file was, and your system has a petabyte's worth of disks full of files and directories? One answer we are sort of falling into right now (based on what I consider to be faulty experience with the World Wide Web) is to push the responsibility for filesystem navigation off fully onto the user (i.e., making them know the full pathname). This is about at its practical limit now. Even if we know "where" it is in the logical tree, the tools available for tree-walking may be insufficient: clients may time out waiting for an answer to something we actually even know. How do we go about discovering which applications and client operating systems that require upgrade? Is it practical to do that? Or, should we be changing the filesystem structure to fix the problem instead of mask the symptoms?

Here are some other areas in the works that may lead toward some future

successes for our problem solving exercise:

- Global File Systems: shared format, out-of-band signalling for locking, cache-invalidation requirement/updating. Cray Research, and others, have done some work in this area.

- File System Emulators: global/proprietary data format internally, represented to each vendor filesystem as "their" own format. Veritas, various AppleShare vendors, and others have made attempts here, some successful, others less so.

- Third Party Transfer: dedicated control server. The client sends a request to a server, which issues a command to the storage. Storage response goes directly to the client. Push technology. Development efforts include those of the Lawrence Livermore Laboratories, IBM, and various IEEE groups.

- Controlled Request: Third Party Transfer without the push protocols. Server returns a command sequence to clients for direct access to storage.

5.1.2 Networks

Networking is, obviously, a critical component of very large scale storage solutions. We can divide it into three areas, LAN, WAN and SAN.

- LAN: Most of the users are on IP-based clients. Their client will use the storage either directly (e.g., NFS-mounted from the "server") or indirectly (user logs into, or otherwise performs work on the server, which deals with disk more directly.) Where clients need direct access to data, LAN bandwidth is an issue. Where servers access data across a LAN, as in the Network Access Storage (NAS) arrangement, LAN bandwidth is a critical issue affecting everyone. It is common to build

a second LAN, out the "back door" of the servers, to attach NAS equipment with higher bandwidth (thus without conflicting with user requirements) without upgrading all the routers, switches, hubs and client host bus adapters to gigabit ethernet. This adds the complexity of making all of your servers into "storage routers", if not IP routers, for getting service to clients.

- WAN: There are two relevant issues in wide area networking, service of data between sites, and mirroring (both for availability/protection, and for performance at the remote sites.) The amount of bandwidth chewed up by the current storage protocols is pretty hefty for what is typically very much smaller-than-LAN inter-site pipes. Extensions in fibre technology now give up to 10 kilometres at gigabit-type speeds, making larger scale storage-over- WAN more practical.

- SAN: Storage Area Networks are a new form of the "back door" second LAN scenario above. It avoids the routing problems, cuts down on server cycles used for storage functions, and significantly increases the bandwidth across that back door network. By putting the right pieces together, storage can be made available outside of the computer room, either through NFS, through extended fibre runs on the SAN itself, or through routing (presumably over IP). A "storage router" can also be configured to include non-SAN equipment into the SAN mix. Unfortunately, it is still the case that "SAN" is a buzzphrase that means something different to every vendor, depending on what they produce.

5.1.3 Storage Hardware

- NFS-served RAID or JBOD: The most common disk service is performed by the Network File System (NFS), over a LAN, using a server with directly attached SCSI disk. In most cases, this disk is in a

configuration typically called "JBOD", for "Just a Bunch Of Disks". Loss of a disk drive means taking the server offline (or at least part of it), replacing the drive, then restoring off tape. While this is not too serious a problem for a small disk served from a user's desktop, it's not acceptable in high-demand, 7x24 environments of thousands of spindles. Most of the major computer vendors produce this type of storage.

Larger sites in particular, but more and more commonly the smaller ones too, are fielding "RAID" in one form or another. See Appendix 1 for a description of the more commonly used levels. RAID can be done either in hardware or in software, or in fact in a combination of them both. Hardware RAID controllers split the incoming data from a single buffer across a series of drives under that controller's "control", with high-speed results. Most do the parity calculations at the controller level, saving the CPU and storage bus bandwidth. Software RAID is done by the CPU, necessitating additional trips up and down the storage bus by the data bits, plus the CPU time to calculate parity, both on writes and reads. The advantages of doing RAID in software are a slight drop in hardware expense (countered by the increase in software cost, but that may be hidden in the fact that you needed the software package for other reasons), and the fact that you can split the data across controllers, towers, or even across sites in various configurations. The protection level can be quite high, but costs rise right along with protection.

- Network Attached Storage: Over the last ten years, there has been a significant increase in the amount of storage connected directly to the LAN instead of to "NFS servers". These devices are typically just NFS servers themselves, stripped and tuned to do nothing else, thus to do this rather well.

Network Attached Storage (NAS), found commonly from, e.g., Network Appliance Corporation and Auspex Corporation, means a standalone machine that is configured to supply storage protocols (NFS or CIFS) over a messaging protocol network (IP-based ethernet) to any clients desired.

The key to these devices is speed. As compared to having a UNIX machine running local SCSI, UFS, then serving via NFS to clients, these boxes have heavily modified or eliminated the UFS portion and most of the kernel overhead, leaving local SCSI, NFS and 100mbps ethernet networking as nearly all that's involved. Some add functionality of significant value, e.g., the "snapshot", where all data (stored by block) can be maintained on line, including the version of a block now obsoleted by user change or deletion, for immediate, non-tape recovery. Snapshots come in very handy for "real-time" backups of systems that must stay live, and will be revisited below.

Compared with the UNIX- or NT-based fileservers they replace, these boxes are blazingly fast, incredibly easy to install, and fairly easy to maintain. The additional heterogeneity has not been, in my experience, a significant problem. What is, though, is scalability. When NAS came out, people thought in terms of gigabytes, scaling up to a few hundred gigs. To grow further, you just add more NAS boxes, more filesystems, more network, and more maintenance. Even if you can handle that in your environment, you still don't easily reach hundreds of terabytes.

- Fibre Channel: The newest form of connectivity, particularly geared to storage scenarios, is fibre channel. It can be over copper or fibre media, has its own protocol stack (upon which other protocols like IP or SCSI ride), and is slowly becoming standardized sufficiently to allow for hope of eventual vendor interoperability. It starts at 100

megabytes per second, bidirectional full duplex, point to point. Second generation, out in some areas now, gives 200 megabytes per second each way. Third generation is 400MBps, and works in many labs now. Later generations are already in progress. The protocol lends itself to routing, aggregating and switching, and looks very familiar to those of us who have some data network administration time. Three typical scenarios arise: point-to-point, loop, and switched fabric.

1. full duplex point-to-point, as with messaging protocol suites like IP. Gives no-collision full bandwidth to the devices on each end, with little overhead. It does not expand well, as the device limit per bus (using a hub in the middle) is 126.
2. loop technology enables setting up separate "buses", which can be connected via routing. Dual loop is a construction where each device has an alternate path, should one loop break or busy out. It allows "double" potential bandwidth while both are working, or slow but steady progress for all when one fails.
3. switched fabric is like the IP or ATM type messaging protocols. Each device "logs in" to the network, and plays as an equal amongst the other devices. Treating devices independently instead of as members of a bus gets past (for many practical purposes) the 126 device limitation, allowing for the hundreds of thousands of spindles we need. Combining switching and dual loop technologies gives the ability to subdivide for management, yet maintain very large numbers of devices with the IP-type switching advantages of minimizing collision.

5.1.4 Storage Software

- NFS:

This is the current technology. Most sites looking at expanding to large storage are already using NFS, and due to it's age and shortcomings, are looking for something more to migrate toward. Some of the problems include:

1. application servers are not efficient fileservers
2. dedicated fileservers have limited scalability
3. NFS is slow
4. server-to-server traffic competes with server-to-client
5. "storage" is more than just data residing on a disk (management issues)
6. impact on users of doing a backup, especially 7x24 shops

Yet, at the same time, NFS isn't going away. NFSv4 is in the works, clients will require some version for the foreseeable future, and it provides functionality we need.

- RAID: Many of the significant benefits of RAID can be obtained via software packages. Unquestionably, in most situations (not all - performance may be too much of an issue for some) there are tremendous benefits to be gained: splitting a write over many spindles is faster than piping it to one, parity and striping in various combination provides for reliability (meaning keeping user data available even when a hardware failure occurs), etc. There are some costs, though. The software is not cheap (you get what you pay for - in my limited personal experience I found it as high a quality software as I've dealt with in any part of our industry), there is definitely a noticeable performance hit both as compared to JBOD and to hardware RAID, and installation and maintenance requirements are large.

5.1.5 Backups

Most larger production sites use some form of commercial software package for handling backups, restores, and often also hierarchical storage management (HSM). Most use tape libraries. The question in front of us now is, "is this sufficient for the future?"

Many software packages were available a couple of years ago. Most are now being swallowed up into the few that can pull off the corporate merger approach. So what is left? Are they worth the time, money and effort?

Then, there's freeware. A large number of small to mid-size sites use free packages such as Amanda (from a group at/around the University of Maryland, along with their Internet Support Group[TM]). I have used it for years, with great success. It, at the moment, does not scale up this big. Given that more sites are growing and that freeware tends to soon do what we need, I would be surprised if this scaling problem doesn't get handled in the near future.

Packages around include Legato (aka NetWorker), BudTool (just swallowed up by Legato), Alexandria, NetBackup (Veritas), Tivoli, and a host of other players with varying shares of the market. Each advertises some slight twist on the same set of themes: they'll roll your filesystems onto tape and keep some sort of track about where they put it. While some do a better job at scaling up than others, I'm not aware of any that will last long the way we are growing. I have not tried Alexandria (just heard stories), but of the other commercial packages my favorite has been NetBackup. In all cases, configuration is a bear, and I've heard of and/or seen Bad Things happening when you suddenly need to restore something.

So, where does that leave us? Assuming all software vendors "fix" their code, do these packages actually supply us what we need? How much time do you have to spend going through indexes looking for "well I think it had a double 't' in the filename" in exabyte-size filesystems?

Tape libraries are still SCSI-based drives, with a SCSI robot. Slowly, the replacements will start coming out, where SAN techniques begin to apply. I have seen (but only in alpha) tape drive striping, fibre channel, and tape-to-disk (Server-Free) backups. These will all help. But no matter what tape technology you select, no matter which drive and robot manufacturer, no matter which software package, no matter how many tape-hanging university students you can dredge up, the questions still remain as to what it is you wish to backup and why, and whether the technology can support your needs.

Fortunately, the "how" has become a bit easier with the "snapshot" concept. Using either hardware (e.g., Network Appliance) or software (e.g., Veritas), one can freeze a moment in time and back it up to tape, without being affected by the fact that users are still making changes both to the filesystem, and to individual files. This needs to be expanded to more sites, and needs to be considered when calculating both your "acceptable window of risk" for creating your backup schedule, and your total number of spindles to purchase.

5.2 Still To Be Resolved

Many technical issues remain on the worklist, some cross over into the "soft skills" area. Some examples:

- disaster recovery. What does it mean? Do we still look at the cost of bringing the whole site back on line? How long would it take? Is off-site mirroring instead the only answer, and if so who pays for that?
- backups. We need to redefine this term, both for ourselves and for users. Is snapshot the technology to use? What does a backup software package now need to accomplish? What sort of tools are required for us to accomplish a backup, under the new definition?

- standardization. In fibre channel, the standards are getting close, but in the rest of the SAN protocol stack, there is still "room for improvement" in the process. "Vendor interoperability varies".

- backup media drive connectivity. This is still SCSI-on-copper. The SAN over fibre channel is "coming soon", but needs integration with all the software packages.

- file locking. The ability to do storage-to-storage data transfers, as opposed to storage-to-CPU-to-storage just isn't there yet. Should be soon, but until the file locking issue is fully resolved, this will be tough. Again, multi-vendor interoperability will be a mudhole. The issue here is that with logical volume management, each logical volume can be made to appear to each CPU on the SAN as though it were local disk. Under the current model, all filesystems (the "volume" equivalent) are assumed to belong to only one CPU, which serves (via, e.g., NFS) disk space to other CPUs. Fully implemented SAN means each CPU and OS thinks it's the sole owner of the same space.

- every machine going onto the SAN will need to be upgraded to an operating system that understands very large filesystems, and presumably a lot more about the new technology. Each will need to be able to drive new host bus adapter cards, at pretty high speeds. Clients, especially those not on the SAN, still need upgrades to their tools and application software to take advantage of the increase in available disk space.

- who installs/maintains. The cost of purchase is bad enough, but the recurring costs will be dramatically higher, in dollars and in people effort, than people are yet ready to commit to.

- fail-over. The state of process fail-over between machines of a High Availability "cluster" of whatever form is still not good. Under some

circumstances it can be made to work, but it is not yet generally applicable.

- overlapping private universes. When someone builds a site this large and expensive, the likelihood is that many other organizations will end up working on it too. Some will wish to share parts of their data with a set of other collaborators, different parts with others, some parts specifically restricted, etc. Other groups will have similar but conflicting needs to share. Currently we have a model of a "shared universe" of CPU time, RAM content, read-ahead buffering and filesystems, where we have only limited ability to control who gets to what. The amount of control we will need exceeds current ACL approaches. Perhaps development of a concept of private universes, with configurable overlapping is required.

6 WHAT IT MEANS TO SYSADMINS

Other than the various SAGE organizations around the world, there isn't anybody out there looking out for our interests. Vendors will push design, standards and development with profitability in mind. Management looks at dollar costs, often limited to the short-term. Who looks at maintainability or "relevant" performance (fast CPUs don't mean fast appearances at the user end)? That falls into our world, the folks who have to "make it work". So, what does all this mean to us? What do we need to be doing, now and as things progress, to ensure that it's "done right"?

First, we need to educate ourselves. The topic area is so large, and of necessity includes so much of what came before as well, that becoming an expert in this sub-field is a daunting task. We as a community need to understand it, then start spreading that understanding around to our individual members.

Next, we need to educate management. Not that this is a new requirement, nor that management is any more ready now than before, but we have an arrow in our quiver this time: the total dollar cost is so large that someone is going to have to sit up and take notice. When we say "this is what you hired me for", they are forced to either listen, or to reject. If they choose the latter, they will soon find out that it isn't a matter of choice, so I'm inclined to think we have a better chance now than in previous situations to have an impact - *IF* we proactively prepare. One area in particular we need to make clear to managers is that we need, even more than before, to be in on the very earliest phases of planning for future business operations, and need to be kept closely in on the loop all the way through to implementation. I have been responsible for more than a dozen full-company type relocations, and in most cases was brought in too late to prevent some obvious and costly mistakes. Building a very large storage site is likely to include either a relocation, or a remodel extensive enough to be equivalent.

We then need to look at re-defining the terms I have alluded to, such as what a backup means, what data means, why do we store it on a disk, how do we present information to users, how we navigate/index/track data locations in storage, etc. Until we, not anyone else, are able to figure these things out, we will be unable to tell vendors what we need. I do not anticipate that individual vendors will be able to disinterestedly come up with these answers on their own. They make money by just selling us more of what we're buying now, and know they will eventually be able to make even more by selling us whatever the world finally comes up with. Any effort they may put into this on our behalf now runs at least slightly counter to their interests (although the argument can be made that being first will be profitable too.) Fortunately many vendors are involved in the process at this point. I don't anticipate that the trade media will be able to come up with answers. I suspect that the track record of "general users" and marketing types will not be broken, and they won't be able to answer these questions either. It

has to come from us, in conjunction with vendor consortiums.

There are a few relevant vendor consortiums now, although each appears to be motivated by a drive to develop the "standard" at the expense of the others. Not a very new occurrence, I'll grant, but still a problem. Fortunately in this case, they've managed to avoid significant divergence thus far, but there is still a long way to go. Nobody but the vendors will have the resources to do the development of new hardware, software, tools, etc. Nobody but sysadmins will have the "pull it all together" requirement, and thus the necessary attitude.

What can we do as individuals? Learn as much as we can. Get involved in organized efforts to build new tools. Get involved with the vendor consortiums to explain our requirements. It will clearly take our efforts, without anyone coming to ask us, for them to reach a real success.

7 CASE STUDY

The Institute for Quantitative Systems Biology (IQSB)

I was called upon to design and implement the computing environment for a new genetics research site. I already managed the "predecessor" site, so had an idea what they were doing and what problems existed in supporting their work. The research facility was to grow a couple of projects at a time, both moving existing groups in, and establishing new groups based on new grants. I expended the ordinary effort to re-establish user requirements definitions, details of which are not relevant here. Conservative projections put the need at a supercomputer plus a petabyte of online data (in use, with any HSM in addition) in four years. Most of the users were more optimistic, and figured two years. Conservative planning meant preparing for either eventuality.

Growth appeared to fit neatly into four one-year iterations, thus I divided

the project into four "phases". Phase 1 is now built.

User requirements for availability dictated maximizing redundancy and high availability (HA) technology. We need to grant various functionality to offsite collaborators, provide for Location Independent Computing (home and on the road), provide significant levels of security (collaborator or granting agency requirements make this different from the average university environment), and be prepared not only for significant growth, but to support projects that may share some resources but not others.

Phase 1 consists of an HA pair of NFS servers (Sun UltraEnterprise 6500s) running the Veritas suite, an HA pair of database engines (Sun UE4500s) with Veritas and Oracle (not the Parallel Server because that conflicted with the Veritas "First Watch" - don't know whether that is still the case with the follow-on product, "Cluster Server"), a backup system (Sun UE4500 and an ATL Products P-3000 16 drive DLT library) with Veritas' NetBackup. Connectivity is provided by a "pile" of Cisco equipment (with gigabit ethernet to the servers and between switches) and Vixel fibre channel switches, the latter handling SCSI between disk towers and servers on some switches, and IP at horrifying speeds on other switches. The 8 terabytes of disk include some Sun JBOD boxes on each server (D1000), and Clariion hardware RAID-5 towers for main storage. There is an eclectic collection of other and sundry workstations, "security" boxes, and odds-and-ends.

The NFS servers are, at this stage, also the main compute engines. They are, using Veritas "First Watch" (soon to be upgraded to Veritas "Cluster Server") fail-over capable for NFS service to the whole site. When a supercomputer is added, these drop back to just doing NFS service. When the file locking problem is resolved, they will supply NFS to non-SAN machines only.

The database pair run separate instances of Oracle upon separate databases. Their fail-over scripts are designed for each to start the other's instance lo-

cally as required. Currently 0.5tb of the 8 are assigned here, but this will likely grow dramatically.

Phase 2 expands the CPU count in the big Suns a little, plus adds a couple of hundred terabytes of main storage. It also makes the "double-router-DMZ" firewall fully redundant, with two separate ISPs. This is the point when we begin mirroring offsite, plus mirroring other sites' data into the local storage towers.

Phase 3 adds a few hundred terabytes, plus a supercomputer (vendor TBD), more SAN switches, and enough more workstations (across multiple buildings) to warrant a bunch more "regular" network switches.

Phase 4 sees more hundreds of terabytes, another couple of tape libraries, more supporting servers, and migration of the NFS servers to another role as we implement the software that makes all servers treat the disk towers as "local disk" (file locking software should be ready by then.)

We have done some preliminary benchmarking. Preliminary, as thus far all of the tests we've run have completed too fast for us to reliably measure. This, beside being the kind of problem you "like to have", has set our testing schedule back some (as has the political reality of a six month delay in getting the Institute doors opened.) When your CPUs are running at very low load, and you want to speed up your overall system, the obvious place to look is the I/O bandwidth and disk access times. That is where this hardware suite shines, and improvements there are showing the biggest bang for the overall buck.

The key is the SAN switch. This box makes each device on the FC-AL think that it and the switch are the only devices, so the contention/collision issues disappear. Yet, there is really only one large loop (configurable - there could have been multiples, but I didn't need them yet), and the "broadcast domain" is consistent with what you would expect (all devices can see all others). Just like a "regular" network. SCSI works at 100 megabytes per

second, one way (with the 200 megabytes per second equipment coming shortly.) Ordinarily, with a bus full of devices, you can't get too close to that. When you switch the bus, you get very close: we are seeing 98+ megabytes per second on those links, and they're dual-connected/full duplex, so we see that in each direction doubled (400MB/s for bidirectional work, which is particularly relevant to an application of software RAID.) Add RAID, and you absorb your data fast at the disk end as well. Add software RAID and for the cost of CPU time, plus some additional bandwidth usage, you can split the writes across multiple switch links to separate towers and separate controllers and shelves, and aggregate writes even further. The vendors are talking about 2.6 gigabytes per second aggregated throughput on this equipment!

We are using some IP across the switched FC-AL as well. At 2.6 GB/s, with the old formula of 10 bits per byte on the wire, that leads to a potential of 26 gigabits per second for "ethernet" between my servers. I use gigabit ethernet across the Ciscos as my "slow speed reserve network"! Of course, I've yet to be able to prove those speeds, but it is quite evident in early testing that this is no ordinary network.

What problems do I face? In addition to political ones, I have all of the classic ones noted in this paper. We have an "appropriate" amount of money going into the initial purchase and infrastructure, although the various tools of our trade aren't all there yet. I don't have enough desk space for enough sysadmins, so I can guess what sort of issue I'm about to face on the support side. I will need a help desk, webmaster and database administrator, whereas I've always had enough "central server support" to be able to take those out of hide. We signed up for the maintenance contracts, but we really don't yet know how long we can afford to pay for all of those. I have 16 tape drives in one library, with a dedicated pile of equipment and cable to get things backed up, but if you'll recall, this is just Phase 1..... I have purchased nothing yet toward the remainder of the project. I have a standard set of

security operations, both host and network, but there are some new areas of concern that haven't yet been addressed, such as the "overlapping private universe" situation above.

8 DESIGNING A SITE

We are the ones who will be called in to build these new large sites, typically at the last minute, after building plans have been completed. We need to look now at what it takes to accomplish this kind of computing environment build. Here are some questions we need to ask, and more importantly, to answer. These questions are either in addition to the normal ones we would ask when building any new site, or are significantly affected by the size of this kind of site, and are geared solely around the large storage problems. Certainly a User Requirements Survey is in order first!

- Above we looked at some trends. Which ones may affect our new site during the next N years, for which we need to plan now? How many years is 'N'? What costs are associated with implementing some of those trends, and who will pay for them (e.g., who pays for, inventories, manages, and provides "approved" software for a pile of Palm Pilots?) Who pays for the implementation of new technology "whizbang" stuff as it comes out, such as wireless quake games over Walkman-style handhelds, especially when it "needs" a connection to the telephony service too?

- What does it take to implement the new technology solution options listed above? Will they actually solve our problems? When might they become available sufficiently to implement?

- What is the level of performance and service required? Are Service Level Agreements in place? Do they make sense given the new kind of environment?

- What are the uptime requirements, the cost of meeting them, the cost of backing off and assuming additional risk of downtime?
- What options are there for "High Availability" or "Always Available"? Can you build in scheduled downtime? If not, what do you need to do in building enough redundancy to ensure service is available while you take pieces offline for repair or maintenance?
- What is the expected lifetime of the installation?
- Is it expandable in large iterations?
- What will it cost to buy? To run and maintain?
- How do you convince the bean counters of the cost requirements? Surely they will understand the basics: 'x' spindles times 'y' dollars/rubles/pesos/ whatever, but how do you explain the cost of GBICs, fibre cabling and cable ties, towers, redundant power supplies, host bus adapters, the need for all those CPUs, the need for site management software that used to be a luxury? How do you explain the large recurring space, power, air conditioning, and consumables (tapes, etc.) costs? I had trouble explaining why the overhead cable tray they selected wasn't sufficient for fibre, and that they needed to buy a more expensive alternative.
- Who will install it? At what cost? How many shifts of how many people to maintain it? Experience level - are they "tape monkeys" or SAGE Level IV senior sysadmins? How do you convince management of the cost of the number of people required to manage this? Are you going to outsource some or all of these tasks? To whom?
- How long will it take to design, acquire, install and configure before you can actually put users on line? This is going to be a sticky point, as it takes MUCH more time than we've needed in the past. Don't

forget the 90/90 rule here (the first 90% of the project takes the first 90% of the time, and the last 10% of the project takes the other 90% of the time.)

- Are the parts you select interoperable and maintainable? What sort of spares kits will you need? Tools, both hardware and software? Don't forget bench space.
- What sort of network support is required?
- Is HSM a relevant technology? It wasn't for us, as the average time between accesses to *every* file was less than 2 weeks, with the longest being 4 weeks.
- How long do you have to change tapes before the next backup starts? (Negative hours isn't a satisfactory answer.) How much will it cost for offsite storage? Who will provide transportation back and forth for all those tapes, what is the lead time required for redelivery, where do you get the muscle-power to cart them back and forth? Where will you store your onsite copies? Where will you store blank tapes? Is your backup schedule well understood? How about your "acceptable window of risk" for files that never make it onto tape based on creation/deletion timing relative to backups?
- Will you "snapshot"? For backup purposes only, or make that available to the users? What parts of the total storage capacity need this?
- Do you need to mirror offsite? Onsite? For backup purposes, or for other sites' convenience? Who pays, if for the latter? What support do those sites require?

9 CONCLUDING THOUGHTS

We face some very big problems. The monster was let out of the box a few years ago, with the massive explosion in Internet connectivity and the coincident significant expansion in "businesses" that shifted much of their operation to some form of "computer". The networking world has been scrambling about for some time to handle bandwidth, addressing and routing problems, but the storage world is just beginning to recognize a need for looking into the issues presented here.

I expect that the computing support world will be overwhelmed by user demand in this area, before we are ready to put solutions into place. I expect that the problems will fall, as usual, squarely on the shoulders of the front-line troops - the sysadmins who have to make it work. It is in our interest to make sure that someone is working feverishly on defining and solving these problems, now, and to ensure that they are doing so in a realistic, implementable manner.

Do I think it'll happen? To quote Harrison Ford's character "Han Solo" in the original STAR WARS movie, "I have a really bad feeling about this."

At least it's Fun Stuff. There is, however, plenty of room for new ideas.....

10 APPENDIX 1

RAID Levels

0 - disk striping. Speeds disk I/O by splitting each activity across several mechanical head assemblies.

1 - mirroring. Provides fault tolerance by making a duplicate, or "shadow" copy of everything written.

2 - parity. Multiple dedicated parity disks, synchronized. Effective for reconstructing large quantities of data. Thinking Machines specific, more or

less.

3 - parity. Additional disk per "set" dedicated to maintaining parity information for a slice of data on the rest of the set of disks. Can be bit or byte striped.

4 - parity, striped by sector or block. RAID 3 at the byte level.

5 - rotated striped parity. Parity data is distributed amongst all disks, as is user data, rather than being segregated to parity-only disks. RAID 3 without the bottleneck of single parity disk. Most common level in use now.

6 - parity plus. Same as 5, but allows for extra parity drives, thus allows for simultaneous multiple-drive failures.

7 - multi-tiered cache. Storage Computer Company proprietary, using asynchronous hardware that shares cache info, block stripes, uses multiple dedicated parity drives.

layered - RAID unit that uses another form of RAID unit instead of individual disk drives as its building block: e.g., RAID 53 is a RAID 5 that uses a RAID 3 unit for each of the "disks" constituting the level 5 unit. Provides potential performance, availability, and capacity advantages.

11 APPENDIX 2

THE PLAYERS

Consortiums and Organizations

FA - Fibre Alliance www.fibrealliance.org

FCIA - Fibre Channel Industry Association www.fibrechannel.com www.fccommunity.org

IETF - Internet Engineering Task Force www.ietf.org = IP over Fibre Channel (ipfc)

SNIA - Storage Networking Industry Association www.snia.org

Examples of Relevant Vendors

- *Network Attached Storage*

  Auspex www.auspex.com

  Network Appliance www.netapp.com

- *SAN Switches, Hubs and Adapters*

  Ancor www.ancor.com

  Brocade www.brocadecomm.com

  Crossroads www.crossroads.com

  Emulex www.emulex.com

  Gadzoox www.gadzoox.com

  Jaycor Networks www.jni.com

  QLogic www.qlc.com

  Vixel www.vixel.com

- *Fibre Channel Disks*

  Clariion www.clariion.com

  EMC www.emc.com

  Fujitsu www.fujitsu.com

  IBM www.ibm.com

  Seagate www.seagate.com

- *Tape Libraries*

  ATL Products www.atlp.com

  Dot Hill www.dothill.com

  Exabyte www.exabyte.com

- *Computing Side Equipment*

  DEC/Compaq www.digital.com

  HP www.hp.com

  IBM www.ibm.com

  Sun www.sun.com

- *Integrators*

  Andataco www.andataco.com

  Storage Technology www.stortek.com

  Veritas www.veritas.com