# Using Gigabit Ethernet to Backup Six Terabytes

W. Curtis Preston
Collective Technologies

# Using Gigabit Ethernet to Backup Six Terabytes

*W. Curtis Preston* – Collective Technologies

## ABSTRACT

Imagine having to prove everything you believe at one time. That is exactly what happened when I was asked to change the design of an Enterprise Backup System to accommodate the backup and restore needs of a new, very large system. To meet the challenge, I'd have to use three brand-new pieces of technology and push my chosen backup software to its limits. Would I be able to send that much data over the network to a central location? Would I be forced to change my design? This paper is the story of the Proof of Concept Test that answered these questions.

### The Challenge

During the design of an enterprise-wide backup system it was announced that the client was purchasing a Sun Enterprise 10000 with six terabytes of data, and an EMC three-terabyte storage array. The E-10000 would provide database, application and file server support. The EMC array would initially be attached to an NFS file server with one terabyte of exported data. Systems of this size would place significant new demands on the backup and recovery system. We needed a much more powerful system then we had initially envisioned!

### The Answer

The capabilities of Gigabit Ethernet, Sun's E-450, Quantum's DLT 7000, and Legato's NetWorker allowed us to accommodate this large server by changing only a few components of the original design. The current system design backs up data at speeds of up to 144 GB/hr. This means that we can back up a 1.1TB system (or E-10000 domain) across the network within an 8-hour backup window.

How does one get a terabyte of data to a tape on the other side of a network within eight hours? The answer is dynamic parallelism. Although several commercial products support this when backing up databases, only three support backing up file systems in this way. (For details on the concept of dynamic parallelism, please refer to http://www.backupcentral.com/parallel.html .) Legato NetWorker was already chosen based on a number of factors, and it supports this kind of backup.

Once we realized the amount of data that would be sent to the backup server, we began to wonder about its capabilities as well. Just how much data can you push through an E-450? The answer, thankfully, was "a lot!" The system is specifically designed for I/O.

Assuming the software can transfer the data and the backup server can accept the data at the needed rate, it needs somewhere to put it as well. DLT 7000s were chosen due to their price/performance ratio, as well as their reliability record with this company.

Since we had to backup terabytes of data across the network, we were going to need a pretty big network! The good news is that Gigabit Ethernet cards and switches just began shipping a few months ago. Our design, originally based on 100BaseT, was quickly changed to accommodate this new technology.

Each system to be backed up by our enterprise backup system is now plugged into a 10/100 or Gigabit Ethernet port on the switch, based on the required throughput for that host. This formed a completely separate backup network, with no route to the backbone.

The end result is a multi-host, multi-platform, enterprise-wide backup solution that uses a dedicated network for all backup data traffic. It is capable of backing up over 1.1 TB in one night. If we need to increase that to several terabytes, all we have to do is use more storage nodes – since the switch is capable of much more than we are currently doing with it.

### History

While consulting at VBC[1], I discovered that they were trying to design a backup system for their entire enterprise. Since this is my area of specialty, I volunteered to be on the project team. The project that followed proved to be one of the most interesting ones to which I've ever been assigned. The first phase was an extensive survey of the user community. Since the client is a very big company, this took quite a while. We found two distinct philosophies within the company.

_____

[1]VBC is an acronym that I will use to refer to the client where all this work was done. While consent was given to write this paper, I find it best to keep them anonymous. In case you are wondering, VBC is short for "Very Big Client."

The network administration group felt that the current network infrastructure was grossly inadequate to handle any sort of network-based backups. They were correct. Some of their servers were still on shared 10BaseT, which can only handle about 300-400 KB/s on a good day! This group felt that the best way to do backups was to do them all to locally attached tape drives on each server.

The systems administration group felt that tape drives put servers at risk. They were also right! Backup processes can sometimes hang on a given tape drive, requiring a reboot of the server to fix them. The older the servers and tapes drives are, the more likely this is to happen. Doing all backups locally would certainly increase the chance that a production server would need to be rebooted before its backups would work again.

I was also told that upgrading the network infrastructure wasn't going to happen any time soon. I was also told that installing private networks was all right – if I made sure there was no routing from the private network to the public network.

### Network Layout

The system we were to design was to handle the backups of several buildings within a five-kilometer radius (see Figure 1). There were two main buildings where most of the systems resided. I'll call them buildings "A" and "B." All buildings are currently connected via a FDDI backbone, but the network within each building is often limited to shared or switched 10BaseT and shared 100BaseT. Building A has approximately 50 servers (mostly Suns) ranging from 1-60 GB. Building B consists of approximately ten servers (mostly Suns) ranging from 1-60 GB – and one 500 GB Auspex. There is also a 500 GB Auspex

in Building C, and five buildings with less than 60 GB of data each.

While most of these servers are Solaris systems, there are a few HP-UX systems and several NT and Novell systems throughout the campus. The Novell and NT systems were not included in the scope of this project. However, we were asked to keep them in mind for future inclusion.

### Design Options

There were four primary options that we considered when designing the backup system: using all local devices, using the backbone, creating a local/backbone hybrid, and a private backup network. There is also a new option that was not available during phase one, but it shows promise and will therefore be included in this report.

**Backups via Local Devices**

This is the traditional method for performing backups. An administrator loads a tape in a local device and some homegrown shell script magically gets the data to tape. As discussed earlier, though, it puts the servers at risk of being shut down to fix tape drive problems. It could be argued this is less likely to happen with modern systems and equipment. It is still more likely to happen, though, if you use the tape drives on the servers for backups. In today's world of commercial backup software, this also becomes a very expensive option. That is because when you use server A's tape drive to backup server B, server A is referred to as a "media server." Most backup software packages charge extra for each media server. This means that it could cost you $5000 or more to use your $2000 tape drive! It's also the least manageable of the options, since you would have to send someone to change tapes all over the place. Installing auto-changers everywhere to fix that would increase the
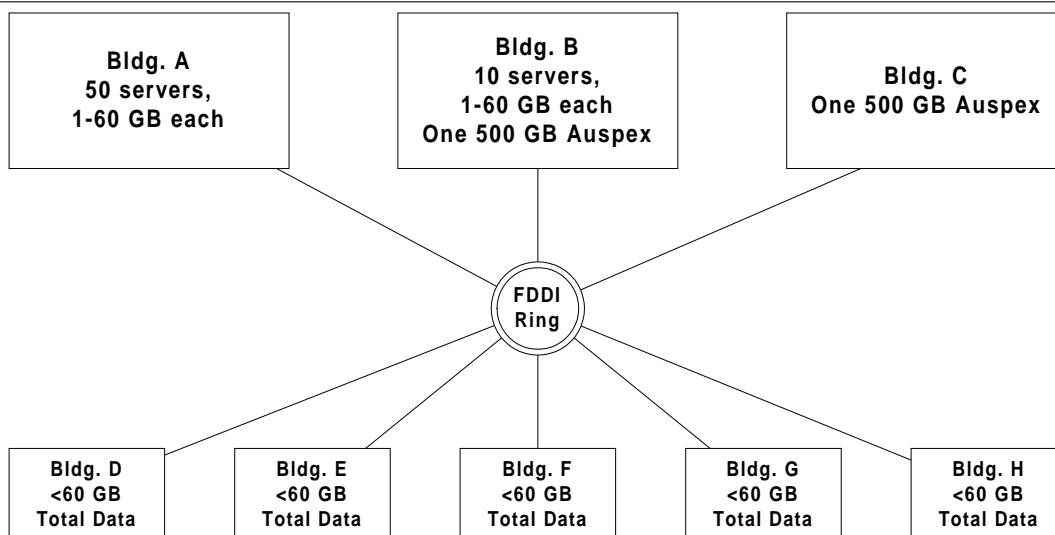


**Figure 1**: Network layout, 5000-ft. view.

cost astronomically due to licensing – another $2000-$5000 per auto-changer.

**Backups via the Backbone**

This is how most backup software packages started. You install a central tape library and backup everything via the backbone. This works fine if you have a large backbone. While VBC's backbone was fine, many servers had only a 10BaseT to that backbone – often shared 10BaseT. This option was quickly ruled out as being impossible.

**Local/Backbone Hybrid**

Can't we all just work together? What if we put tape drives on the really big servers, and backup the smaller clients across the backbone to those servers? This is used quite a lot in companies across the world because it allows for a great deal of flexibility. However, it has "hidden'" problems. First, consider all the problems listed under "Backups via Local Devices," and the small amount of available network bandwidth. Even if we had more bandwidth at our disposal, we will soon outgrow it.

However, the most important problem with this design is one that does not immediately come to mind. Since you are attaching the devices to your "biggest" servers, you are probably also attaching them to your most important servers. Not only are you putting servers at risk, you are putting your most important servers at risk. Also, these servers were designed to do the job they were made to do – not to do backups for you! Causing each large server to also become a backup server puts additional load on your most important servers. This additional load will mean slower response times for the customers trying to use these machines.

Another problem with this design is the load created by restores. Many people design their backup networks without thinking about restores. Suppose for a minute that your network and servers are very slow at night while your backups are running. Most of the issues above do not apply to you. Now ask yourself what will happen when one of the servers dies during business hours? Your production server is doubling as a backup server, so it will have to slow down to respond to your restore request. Your network will come to a screeching halt while you transfer data from the backup server to the server that needs to be restored.

This option was rejected for the reasons listed above.

### Private Backup Networks

What if you created a special network just for backup traffic? You would then be allowed to backup and restore any time you want. You could put the backup devices anywhere you want, and the only load on a server would be to put its own data on that network. (You will see later that this is actually the design's only limitation.) In our particular environment, this would mean installing a separate network in each building, sizing each network by how many servers are present. Systems in each building would then backup to a local "media server" within that building, keeping all backup traffic off of the backbone.

### Storage Area Networks

This is "bleeding edge" stuff, but promises to bring great things in our future. The one limitation of the "private backup network" method is that each server is responsible for transferring its data to the backup server across the private network. What if you have a very small server (e.g., an Ultra 2) that happens to have terabytes (or even petabytes) of data on-line? A good example of such a server would be an imaging server. It may contain thousands of images, but only a few of them are asked for at any one time. Its primary job is to keep track of these images. Because of this, the computer in front of the imaging system does not have to be extremely powerful. It just needs to be able to address all the storage. The end result is that you have a tremendous amount of data sitting behind an extremely small server. That server might not be powerful enough to transfer all that data over the network, no matter what kind of bandwidth it has available.

The answer is a storage area network, or SAN. It is beyond the scope of this paper, and we did not choose it because it was too bleeding edge at the time. Imagine if you had a separate, SCSI based network that contained only your storage devices. This SAN would have at its center a "SCSI switch," connecting a central storage device (such as a large RAID array) to the systems that needed to access that storage. Every system can see the disks as being locally attached and can access them at Ultra SCSI (40 MB/s) or Fibre-channel (100MB/s) speeds. A backup server could also be connected to the SAN and access every servers disk drives locally. That way it could backup a client's disks without having to use the client's CPU at all. The only problem with doing it this way is that you have one system writing to the disk and another one reading from it. The administrative overhead of resolving these issues caused us to reject this method for now. It may be more viable in the future as these issues are resolved.

Another way to use a SAN is to locally attach a large storage device (such as a tape library) to every server that needs it. This is the way that Legato's new "Smart Media" program works. It cooperates with the SCSI switch to dynamically allocate the entire library to every client while that client is trying to backup. This would work very well with the E-10000 where you have both a large server and a lot of storage attached to it. This would allow you to use the shared memory type of backup that greatly reduces your CPU load while still allowing shared access to the tape library! The only problem with this method is that it

was unavailable when we started the project!  It was released September 1, 1998.  It promises to be quite interesting.

### Chosen Design

We decided to go with the private backup network design.  It is simple, and allows for a great deal of flexibility.  New clients can simply be plugged into the network and added to the configuration.  They will not need to have tape drives installed on them – only the backup software.  Backups and restores can happen any time without impacting anyone but the servers that are being backed up or restored.  Since switches are so inexpensive now, it also turns out to be the least expensive option.  VBC also has over 100 of Sun's combination SCSI/10BaseT cards in use, which means that most of the servers will not even need to buy a NIC to be added to the backup network.  (This is probably the case for many Sun shops, since these cards have been widely sold over the past few years.)  Logically, the network will look like the one in Figure 2.

### Two New Challenges

For a lot of reasons external to the project, significant time passed between the preliminary design phase and any sort of design testing, pilot or implementation.  During that time, two very large projects were introduced that would require a significantly more powerful backup system than the one described above.  However, you will see that we actually decided only to change certain pieces of the design.  The actual overall logical design stayed the same.

As you read this section, please do not think that it does not apply to you because you are not going to purchase an EMC array or a Sun E-10000.  The problems apply to any shop with a single server that has close to (or more than) a terabyte of data.  Backup software does a great job of backing up hundreds (or even thousands) of small or mid-size computers to a central location.  However, when one single host becomes very large, the challenge becomes quite different.
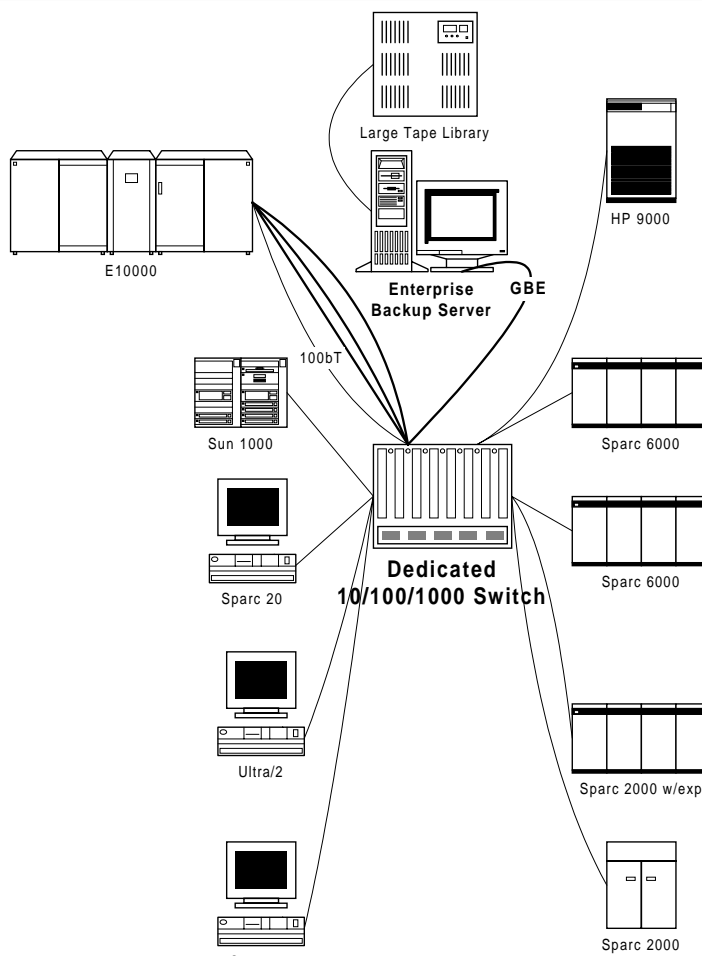


**Figure 2**:  Dedicated backup network design.

## Sun Enterprise 10000: A System That Doesn't Share

The Sun Enterprise 10000 is the result of some technology purchased from Cray when they merged with SGI. It is a very large, multi-processor, multi-host, single-backplane system. It consists of a central unit with a backplane in the center of the unit, which is why Sun calls it a "centerplane," which can accept up to 16 I/O boards. Each I/O board can handle up to four CPUs, 4 GBs of RAM, and 4 SBUS cards. That means that the system supports up to 64 CPUs, 64 GBs of RAM, and 64 SBUS cards!

The E-10000 can then create up to 8 "virtual hosts," (referred to as domains) each of which has their own boot drives and runs their own copy of the operating system. You can then dynamically allocate the I/O boards to these domains, while they are currently up and running. This allows you to allocate more CPU and memory resources to an important, cyclical task, such as end of the month (or year) accounting. Once those resources are no longer needed, you can then deallocate those resources and allocate them for some other purpose.

Although this sounds like a beautiful system, the problem it presented to us was simple matter of arithmetic. VBC was planning on having anywhere from 250 GB to 1.5 TB of storage connected to each of these domains. If we used the current backup network design, the biggest pipe we would have available would be 100BaseT. At maximum speed, a 100BaseT client can back up 288 GB in one night. (That's represents saturating the 100 Mb/s pipe. 10 MB/s * 60 seconds * 60 minutes * 8 hours = 288 GB) We would need a system that was approximately six times faster than that!

The E-10000 does offer other alternatives, which are similar to the others discussed earlier – with a slight twist. Remember that each domain is truly a separate host, and that means that you cannot transfer data via the centerplane from one domain to the other. You have to use the network like anybody else. However, there are some interesting options that are presented because of the unique architecture of the E-10000.

The first option is that one could mount a large jukebox off of one of the I/O boards and dynamically allocate it to each domain during the backup cycle. This would require extensive use of the Dynamic Reallocation feature of the E-10000, which is obviously completely new and very "bleeding edge." It would also present some interesting logistical challenges for the backup software. Imagine a backup jukebox that dynamically disappears and reappears on different media servers! How would you handle the inventory of such a beast? For this, and other reasons we rejected this idea.

There is also one other really new option if you are using Legato NetWorker. They have a new module referred to as "Smart Media." It allows you to connect several hosts to one library via a SAN switch (discussed above). These multiple hosts could, of course, be multiple domains within an E-10000. Legato would understand that all of these hosts have full access to the same library and dynamically handle the reassignment of it to the different hosts who need it. Only one host will be able to use the library at one time, however. This option was not available when we started the project, and we also believed that we would like to keep devices off the servers if possible.

And one final option should also be considered if you have purchased an E-10000. The current revision of the E-10000 does not support data transfer between domains, except by using the network like any other machine. Future revisions may allow this. This means that you could have one domain act as a media server, and have the other domains send their data via the centerplane to that domain. This option is currently unavailable and will have its own unique set of problems, but it's something to consider in the future.

### Terabyte NFS Server

The second challenge that came about was an idea to create an NFS server that would service multiple departments, totaling about one terabyte of data hanging off of a single system. This was actually a worse problem than the one created by the E-10000. The reason is that our design called for using each client's CPU to transfer the data to the backup server via the dedicated backup network. Based on some initial tests, we were using every bit of the E-10000's resources to transfer data at gigabit speeds. The problem with this configuration is that the system that was going to drive this terabyte file server was only an Ultra 2 with two CPUs and 512 MB of RAM. This is about one eighth of what we were using on the E-10000! I believe that this system will require us to split its full backups across multiple nights. For a discussion of why we were trying not to do that, see "Single Client Backup Throughput."

### New Limitations

**Single Client Backup Throughput**

It is important that we be able to do a full backup of any given client in one night. The reason is a feature in NetWorker called the "All Saveset." Using the All saveset automatically backs up any file systems present on the client. It requires zero administration once you configure NetWorker to use the All saveset, since it will automatically discover that you've installed a new file system and back it up. However, the one limitation of the All saveset is that you must back up the entire client each time you run a backup. This is not a problem on the nights when you are performing an incremental backup, but when it's

time for a full backup you need to be able to fit it into a single night. Splitting up a client's full backups across multiple nights means not using the All saveset and manually specifying the file systems to back up. This requires that you then monitor the client for any new file systems as they are created.

The original design was based on 100BaseT, allowing a maximum transfer rate of about 10 MB/s, which allows us to perform a full backup of a 288 GB client in an 8 hour period (10 * 60 * 60 * 8). Please note that both the terabyte file server and at least one domain on the E-10000 are approximately four times that size. 100BaseT was not going to do the job.

**Backup Server Throughput**

The backup server would obviously also need to be able to accept more than 10 MB/s of data, allowing it to back up more than 288 GB per night, or we'd never get the job done! We would need a more powerful network, and a more powerful backup server to

connect to it. One potential solution would be to install a second backup server. Another solution would be to increase the pipe.

**Storage Capacity**

Storage capacity would also have to be drastically increased to accommodate the needs of these new very large clients. The original jukeboxes were going to hold somewhere around 52 tapes. That would barely provide enough capacity to backup these new systems one time!

**Final Design**

We wanted to stay with the same logical design, because we believed that keeping tape drives off the servers was a good idea. So we decided to just beef up the subsystems of the original design. 100BaseT became Gigabit Ethernet. The Sun Ultra-2 became a Sun E-450. And the tape library went from 52 tapes and four drives to 496 tapes and eight drives! What
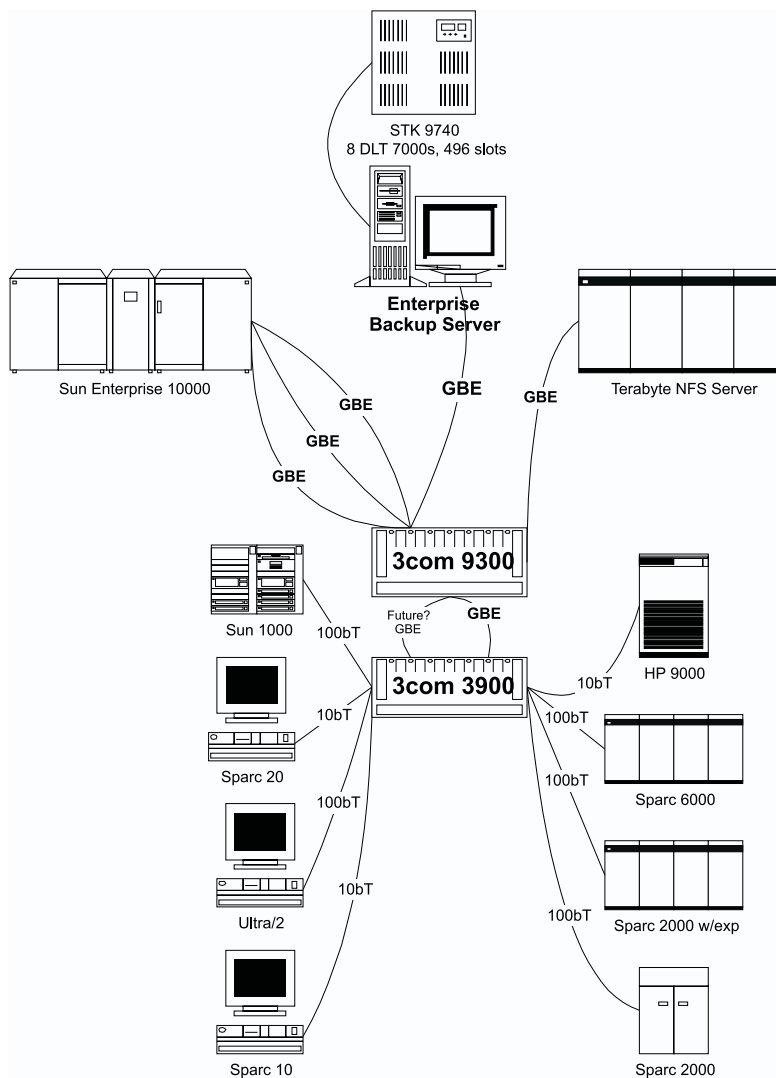


**Figure 3**: The final configuration.

we didn't know until we tested it, though, was whether or not we could scale this far with Legato NetWorker, and whether or not Gigabit Ethernet would actually be that much faster. First, here are some details on the pieces of the new design.

We went for a combined approach on the network. We would provide 10/100 to those clients that needed it, which would be the bulk of them. We would then hook that switch to a Gigabit Ethernet switch via a Gigabit Ethernet uplink. The switches that we chose to do this were the 3com 3900 and 9300. The 3900 has 36 10/100 ports and from one to three Gigabit Ethernet links. See Figure 3.

### How Well Did it Work?

It worked well enough to get the job done, but not as well as I had hoped. My current opinion is that Gigabit Ethernet may be ready for switch-to-switch communication (and it is), but the servers are just not ready to push that amount of data out an Ethernet port – given Ethernet's inherent limitations. You can actually get about the same throughput with a quad fast Ethernet card and Cisco's Etherchannel or 3com Trunking software.

I will say a few things in Gigabit Ethernet's defense. First, I believe that it really caught the OS vendors by surprise. I had always heard of Gigabit Ethernet as a backbone technology, not something to be directly connected to a server. Who would have thought that crazy people like me would want to actually use it to sustain a transfer rate of 500 Mb/s while backing up a server? It's also important to note that the standard wasn't really ratified until just a few months ago. Also, the main Achilles' heel of Ethernet (when running at Gigabit speeds) seems to be its 1500 byte frame size. A frame size of 9000 bytes is much better suited for the task and NICs and switches from

Alteon support this frame size. They've also proven that by using that frame size they can more than double the throughput from a single server while simultaneously reducing the CPU load by 50%! So I wouldn't give up on Gigabit Ethernet just yet.

### The Numbers

Since the ultimate goal of this private network was to backup the data using Legato NetWorker, most of the benchmarks were made with that same software. After getting less than optimal results, we did some other tests without it to see if it was causing the problem. The main system that we used for testing was an E-4000 with 8 CPUs and 1.5 GB of RAM. The disks we were reading from were in multiple Sun A-5000s striped for optimum speed using Veritas volume manager.

The first tests were done with NetWorker itself. By using different combinations of client, device and server parallelism, we tried to stream as many drives as possible. (We defined streaming as continuous operation at close to 10 MB/s.). Since not streaming a drive is neither good for the drive or the tape, our goal was to find the smallest number of tape drives that could achieve the highest overall throughput.

The next test was designed to determine the maximum throughput of Gigabit Ethernet. We did this by creating a file that was big enough to be significant, but small enough to fit into RAM. We then read that file into memory by copying it to /dev/null. Then we used rcp to copy it from one system to another system's /dev/null. This would result in minimal disk operations and hopefully the fastest overall throughput.

We also simulated NetWorker by issuing multiple rcp commands simultaneously, copying several large files from one system to another system's disk.

| Tool | Number of Devices written to | Threads from client | Total throughput | Average throughput per device | CPU Util On client | Percentage Memory Used |
|---|---|---|---|---|---|---|
| NetWorker File System Backup | 1 | 4 | 10_MB/s | 10_MB/s | 40 | 30 |
| " | 2 | 8 | 20_MB/s | 10_MB/s | 80 | 80 |
| : | 3 | 12 | 25_MB/s | 8_MB/s | 100 | 100+ |
| Copy of file in memory | N/A | 1 | 15_MB/s | N/A | 10 | 50 |
| " | N/A | 2 | 30_MB/s | N/A | 60 | 70 |
| Multiple rcps | N/A | 5 | 40_MB/s | N/A | 80 | 100+ |
| " | N/A | 6 | 39_MB/s | N/A | 90 | 100+ |
| Multiple tars to tape | 4 | 4 | 20_MB/s | 5_MB/s | 90 | 100+ |

**Table 1**: Backup throughput.

We then used multiple simultaneous tars to get even closer to what NetWorker was trying to do. The results of these tests are in Table 1.

There were many tests performed at different times with different amounts of memory, etc. This table represents a summary of the results we found. Pushing a Gigabit Ethernet connection at anything even close to 40 MB/s (320 Mb/s) required every bit of memory that we had. At first, we thought that it was also a CPU limitation, until we ran the same tests on a much smaller system – an Ultra 2 with 1 GB or RAM. The Ultra2, with about 25% of the CPU power that the E-4000 had, ran out of steam at about the same place. We believed, therefore, that the high CPU utilization was due to a very excessive scan rate. The CPU was spending all its time freeing up memory and very little time doing the actual job we asked it to do!

We discussed using 3 or 4 GB of RAM to see if we might have gone a little faster, but we didn't for two reasons. The first was that this was not going to be the configuration of the E-10000. The second was that we didn't have it!

### Implementation Issues

This project is a large one, and we are currently in the Pilot phase. The goal is to back up one of several buildings with this configuration, and then to roll out this design to other buildings. Now that testing is out of the way and we know the capabilities and limitations of this system, we can devote time to resolving implementation issues.

### Disaster Recovery

The system is setup to perform a full backup once a month and a level 5 backup once a week of every system. The NetWorker indexes also receive a full backup every day to one tape. These backups are then automatically cloned (copied) and sent offsite as soon as they are made. This offsite storage of essential data will form the basis of the disaster recovery plan.

Currently, level one and level two disasters are going to be handled by using servers in other buildings to recover essential data. There is another project to incorporate a disaster recovery service in the future. We felt that establishing a solid foundation of backups being sent off site was the first step towards making such a service valuable.

One type of small disaster that is not addressed in the pilot (but is addressed in the overall design) is the failure of the tape library. A second library in another building will be added in phase two, and it will become a storage node (remote device server) for the server in this building. If this jukebox fails, that jukebox will automatically take over!

### Media Management

Several systems have been put into place to automatically:

1. Clone certain types of backups
2. Import and export the appropriate tapes to and from tape library
3. Separate full and incremental backups to allow separate retention periods of those types of backups

There are also procedures that dictate daily operations to ensure that enough tapes are available to the library at all times, as well as how to load special tapes for a restore from old tapes. All tapes that are not stored in the library will be stored in Wrightline media cabinets, customized for the types of media we have. There will also be a media database that will track the location of all tapes at all times. A dedicated operator staff will handle the physical management of all tapes.

### Report/Error Notification

There are mail aliases already defined for backup success/failure notification, and this system will use those same aliases. Every backup sends an email message with the subject line of "Successful backup of $GROUPNAME on $HOSTNAME." If any failures are detected, this subject line is changed to "FAILED backup of $GROUPNAME on $HOSTNAME." This allows those reading the mails to easily notice which backups had problems.

The NetWorker bootstrap report is handled in a special way. It is:

1 Saved into a log file in /nsr/reports
2 Printed to a special printer with a header specifying to have it given to the appropriate SA
3 Emailed to the backup notification list
4 Emailed to a few external email addresses so that it will be available even after a level two disaster

### Index Management

One of the requirements of designing such a system would be to estimate the size of the index, or database, that keeps track of all the backed up files and their locations. The size of this index is essentially 225 bytes times the number of times that each file is stored in the index. We first needed to decide on our backup schedule, since it determines how many copies of each file will be stored in the index. Our decision was to perform a monthly full backup, followed by a weekly level 1 backup and daily incremental backups. Also, by looking at our current backup logs, we estimated that 3-5% of all files would change within any given month. We also decided to keep three months of "browse" information online. At any given time, therefore, there would be approximately four to five references to each file stored in the index; we added one more for good measure. You then need to determine the number of files in your environment. We did this with the find command. This brought our

estimate to 13.5 GB[2]. This will obviously need to be monitored over time for growth and performance.

## Summary

The current design accomplishes the goal, even though it is not as fast as we originally hoped for. The only area for concern is that if any single system grows well beyond a terabyte, we will have to give up the "All saveset," and spread its full backups over two nights – something that we really don't want to do. It may then be necessary, to make the system go faster than it currently is capable of. If that becomes necessary, two possible enhancements could be explored.

The first would be to "go with the flow." A lot of work has been done in the last few years to make local backups faster with a much smaller CPU load than if they are performed over the network. This design ignores all those advancements because we are doing all our backups over the network. We did this for a reason, but perhaps putting backup devices on the large clients is the only feasible method once they approach a certain size.

The second possible enhancement would be to evaluate the Alteon (http://www.alteon.com) switches and NICs that support the "jumbo frames" of 9000 bytes. Alteon is claiming that they support a much faster transfer rate over Gigabit Ethernet with a much lower CPU usage. If that's true, the system could become over 100% faster just by changing which network we use.

## Acknowledgments

I would like to acknowledge several people for their contributions to this project. Jeff Rochlin: for his input into the original design; The system would have looked much different without it! Rob Cotter: for his input into the final design and the chance to prove and implement the system. Celynn, Nina & Marissa Preston: for their understanding while I travelled to make this happen.

## Author Information

W. Curtis Preston has six years of experience in systems management and is currently a Principal Consultant at Collective Technologies. He specializes in designing and implementing leading edge enterprise backup and recovery systems for both small and large companies. Curtis speaks about this topic around the country, and this is his second talk at LISA. He is also published in SysAdmin magazine, and is currently completing a book for O'Reilly and Associates. The book's working title is "Essential Backup and Recovery." Updates on the book (and other backup resources) can found at http://www.backupcentral. com . Reach him at curtis@colltech.com .

---

[2]10,000,000 files × 6 copies × 225 bytes = 13.5 GB