SAGE
THE SYSTEM ADMINISTRATORS GUILD

ISBN 1-931971-02-1

9

W. Curtis Preston & Hal Skelly

Backups and Recovery

SAGE

9

Short Topics in
Systems Administration

Rik Farrow, Series Editor

Backups and Recovery

W. Curtis Preston and Hal Skelly

SAGE
THE SYSTEM ADMINISTRATORS GUILD

About the Series

This is the ninth in a series of booklets that SAGE is presenting to the system administration community. They are intended to fill a void in the current information structure, presenting topics in a thorough, refereed fashion but staying small enough and flexible enough to grow with the community. Therefore, these booklets will be "living documents" that are updated as needed.

Series Editor: Rik Farrow

#2: A Guide to Developing Computing Policy Documents
Edited by Barbara L. Dijker

#3: System Security: A Management Perspective
By David Oppenheimer, David Wagner, and Michele D. Crabb
Edited by Dan Geer

#4: Educating and Training System Administrators: A Survey
By David Kuncicky and Bruce Alan Wynn

#5: Hiring System Administrators
By Gretchen Phillips

#6: A System Administrator's Guide to Auditing
By Geoff Halprin

#7: System and Network Administration for Higher Reliability
By John Sellens

#8: Job Descriptions for System Administrators, Revised and Expanded Edition
Edited by Tina Darmohray

#8: Backups and Recovery
W. Curtis Preston and Hal Skelly

About SAGE and USENIX

SAGE, The System Administrators Guild, is a Special Technical Group within the USENIX Association dedicated to advancing the profession of system administration.
USENIX is the Advanced Computing Systems Association.

9 Short Topics in System Administration

Rik Farrow, Series Editor

Backups and Recovery

W Curtis Preston and Hal Skelly

Published by the USENIX Association for
SAGE, the System Administrators Guild
2002

 ISBN 1-931971-02-1

To purchase additional copies and for membership information, contact:

The USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA USA 94710
Email: office@sage.org
Web: http://www.sage.org

First Printing 2002

Printed in the United States of America, on 50% recycled paper,
10–15% post-consumer waste.

Contents

1. Overview

Before beginning a discussion of backup and recovery, let's look at why we even go to the trouble of making backups.

Why Do We Make Backups?

System administrators must do many tasks, including installing, configuring, and maintaining system hardware and software, adding and deleting users, monitoring disk and CPU utilization, tuning for performance, educating users, ensuring security—and making backups. The job of creating backups is often given to the most junior administrator, because it can be very repetitive and not very glamorous. Backups also consume valuable time and resources. Why suffer through all of that work?

For one thing, it doesn't take long in the IT world for things to break. Hardware fails, software becomes corrupted or has bugs, and users make mistakes. There's an old saying, "There are two kinds of motorcycle owners: those who have fallen and those who will fall." The same applies to system administrators: There are those who have lost data and those who will lose data. Preparing for "normal" failure is one good justification for backup.

What about a disaster, such as a tornado, flood, hurricane, earthquake, or fire? Next time you're in your computer room, look at the ceiling. Does it have sprinklers? What will happen to your machines when the sprinklers go off? Are you storing your backups in the same room? What will happen to those tapes? What is your recourse when the air conditioning fails on a weekend, server performance slowly degrades, and disks or equipment fail in non-deterministic ways?

As disastrous as the events of 9/11/2001 were, they reminded those of us in the computer business that we also have to prepare for *man-made* disasters. The number of hackers and crackers trying to get to your data has been increasing steadily. Other dangers include acts by malicious ex-employees and really egregious user mistakes (e.g., *rm –rf .*, or DEL /F /Q C:*.**).

Administrators may also face external requirements for record retention, such as a state or federal agency that requires you to keep documents of a certain type for *n* years.

It is your job to be able to recover from any type of problem and return your systems to full functionality. Just as we all carry insurance for our houses, health, cars, and life, we should make sure we've done what we need to do to be able to recover our sys-

tems in time of disaster. This, of course, requires a disaster recovery plan. One of the essential pieces of a disaster recovery plan is a solid, well-tested backup and recovery system. The remainder of this booklet focuses on creating such a system.

Backups and Disaster Recovery

The following six principles should be considered when designing your backup and recovery system:

1. Define (un)acceptable loss. Determine how much you will lose if you don't have a backup. That will help you decide how much time, effort, and money to spend on protecting that data.
2. Back up everything. You have to make sure that all information is backed up, including data, metadata, and the instructions you or your successor will need to do the recovery.
3. Organize. You need to be able to access the correct media for a recovery, as well as the documentation that explains your disaster recovery plan.
4. Protect against disasters. All the steps you take to safeguard your process must be disaster-proof, including guaranteeing tape security, keeping spare tape drives, and storing copies of your documents off-line.
5. Document. You need to have all the steps for recovery well thought out (and tested) ahead of time so that you or your delegate can perform a recovery correctly even from a very old backup.
6. Test continually. Your plan is only a proposal if it has not been tested. You must prove that the process will work.

(Un)acceptable Loss

If you are like most auto owners, you will have (or must have) auto insurance. Look at the deductible amount. This is the amount of money you can expect to pay in the event of an accident, before the insurance company pays any money on your behalf. The higher your deductible, the less you pay monthly for insurance.

This concept applies to backups as well. The more you spend on your backup and recovery system, the less you will lose when disaster strikes. The less you spend on your backup and recovery system, the more you'll lose in time of disaster. How much of your system's data can you or your company afford to lose?

The first step here is to classify the data. Some data is easily recreated, while other data is irreplaceable. A software development project may be the sort of data that can be recreated. A financial transaction with a customer can almost never be recreated. Generally, if data is being created by a single person or group of people, *without interaction from anyone outside your company*, then that data is probably recreatable. That doesn't mean it should not be backed up, though. It means that you don't need to take backups every 20 minutes to ensure that every byte is captured. If data is not recreat-

able, then it needs to be backed up constantly, usually using transaction logs in a database.

Assign a monetary value to the data. For example, how much would it cost if five developers had to totally recode and debug a week's worth of work? How much would it cost your company if a day's worth of interactions between your customers and their customer support representatives were lost? Some data may have indeterminate value depending on the viewpoint of the evaluator. What if your bank lost your paycheck? It would be devastating to you, but fairly insignificant to the bank (except for their credibility, which is a highly valued product). Obviously, you don't want to spend $100,000 to protect $10,000 worth of data. You do need to consider the liabilities that your company will incur if you lose data. In today's world, of course, you must include the impact data loss will have on the image of your company, since many system outages are reported via the Web—or the evening news.

Back Up Everything

Compared to other files on a host, operating system files will change very little. With the exception of patches, the files are very stable, so you may be tempted to save some backup media space by not backing up system files. However, computer systems suffer from entropy as well as all other processes in our universe. Therefore you should back up virtually everything, to be sure that you have captured those changes. (Of course, you can exclude lock files, temp files, core files, and other unimportant files.)

Backing up everything except what you are positive you will never want is a lot less risky than using include lists that you create and manage yourself. Such lists must be manually updated each time you add a new drive or filesystem to your network. If you don't remember to do so, or if you make a typographical error when entering a new disk name, the new drive will never get backed up. This is why it is much better for your backup system to automatically discover which filesystems and drives it needs to back up.

While you are backing up your files and databases, don't forget to back up your backup software. Many products keep a database, log, or index of the backups they perform, which makes finding a particular file a much less arduous task. However, the backup of these indices then becomes the most important one in your system, since you can't recover any of your other backups without it. Make sure that recovery of your backup software indices is the easiest and most tested in your entire environment.

It's also important to back up your system configuration, including any metadata associated with your volume manager. On UNIX systems, Veritas's Logical Volume Manager, Sun's Solstice Disk Suite, and AIX's LVM all have configuration databases that need to be saved. Windows 2000 and later also uses the Veritas Volume Manager. These configuration databases are crucial to rebuilding the logical volumes that are increasingly important in managing the storage attached to your servers. That being said, it is also a good idea to keep a printout of these configurations so that you can quickly recreate a damaged system from new disks.

Organize

The more standard your configurations are, the easier they are to replace. A few suggestions:

- Keep the system disk on a single device or volume. Spreading the operating system over multiple disks makes recovery much more difficult. It also helps if all operating systems are configured similarly—kept at the same patch or service release level, for example.
- If possible, keep all system disks the same size. This will make replacement much easier.
- Disks that serve the same function should be partitioned the same way. For example, it helps if you always install databases (e.g., Oracle or SQL Server) on a separate disk, and those disks are always the same size.

You will also need to keep track of your backup volumes (the media). You need to be able to find any piece of media reliably and quickly. I'm sure we've all seen situations where small mountains ot tapes were piled in cardboard boxes in an unused cubicle. Don't let that happen to you:

- Give each piece of media a unique external identifier label.
- Use a database to track volumes and locations.
- Use bar-coded media: many of the newer robotic devices can read these codes directly.
- Be consistent in the manner and location in which you store tapes. They should all be lined up in the same direction, label visible, in a secure but accessible location.
- Make sure the storage area is temperature- and humidity-controlled.

Keep online copies of all of your documentation. Your plan is an evolving document. The easiest place to maintain this dynamic information is online:

- Use tools such as SysAudit or sysinfo to save information about each server.
- Keep both hard and electronic copies of your procedures up to date.
- Save the information to CD-R or other removable, random-access media.
- Create a tar file of the information.
- If you are placing critical information on removable media, include a way to access the information (e.g., gnutar, WinZip, gzip, Acrobat Reader).

Protect Against Disasters

Protect your documentation as well as your backup media. The authors live in earthquake-prone California. Many risks are associated with a major earthquake: fire, building collapse, infrastructure collapse (severed phone and network lines), physical damage of equipment, loss of electrical power. Ideally, you should make copies of your backups, storing one duplicate onsite and one offsite. Ideally, offsite storage should be done every day. How often you actually send tapes offsite should be determined by

your calculations of acceptable loss. Inany case, you should visit your offsite storage regularly to check the condition of your media and to spot-check availability of a random subset. The offsite storage site needs to be far enough away from your production site that the two locations would be unlikely to be affected by the same disaster.

Document

Take that old advice, "Get it in writing." If possible, produce documents in a portable format such as HTML or PDF. Keep copies distributed on a small subset of hosts so they will be available, even if multiple systems are damaged. Also keep printed copies stored both onsite and offsite, in case of a complete disaster.

Test Continually

All the successful backups in the world will avail you nothing if the restore does not work due to a media error. The only way to ensure functionality is to run test restores under various conditions. A good test is to hand your documentation over to the Network Operations Center and see whether they can complete a recovery without paging you. Another important point to test is whether your tapes can be used in a different drive or robot from the one they were created on. This will require that your drives and libraries be maintained and cleaning tapes used when required. Finally, test your disaster recovery plan from end to end on a regular basis—quarterly or semi-annually.

The What, When, and How

The previous section contained an overview of how to create a disaster recovery plan. This section goes into more detail about Step 2, "Back Up Everything."

What to Back Up

What are you going to back up, the entire system or carefully selected filesystems? Is there data in places other than filesystems (e.g., raw partition databases)?

A system administrator must determine what is important in the environment, which is a very complicated and subjective evaluation. The HR files may not be important to the scientist (until she wants a raise), and the database of equipment may not be important to the departmental timekeeper, but all the bits and bytes in a computer system are important to someone. So rule number one is **Back everything up!**

Envision your worst nightmare disaster. Be pessimistic, along lines like these:

- The company loses $1 million every hour the computer system is offline.
- The UPS overheats and starts an electric fire that crisps your central file server.
- The backup tapes from the most recent backup were destroyed in the fire.
- The telephone number for the offsite storage vendor was on a piece of paper next to the system and is now just a bit of ash on the floor.

This kind of scenario helps indicate all the types of things that need to be backed up. Make backups of backups. Make backups of documentation. Make backups of

configuration information. Make backups of the backup logs and catalogs. Have an alternative method for restores (e.g., a portable standalone tape drive if you are using tape backups). Make copies of your system configuration information (disk layouts, volume management, database setups, license keys). Become involved in knowing how the systems are configured so that you know ahead of time that server X was the main SQL Server, and how the database information was distributed on the disks.

Are you sure you're backing up everything? You will often see someone pose a scenario like this on a mailing list:

1. We back up all drives from the list every night.
2. The backups complete without error.
3. The configuration was changed and we decided to check for the last full backup of drive X.
4. There is no indication that drive X was ever backed up!

All backup products, in some form or another, provide a method for you to specify what to include for the backup. You can approach this two ways: either tell the backup software to back up only what is in the *include* list, or tell it to back up everything except what is in the *exclude* list. If you take the latter approach, you will never miss backing up something when the system changes (e.g., when you add a disk or volume). If you rely on an include list, you will almost certainly miss something sooner or later.

Many of the freeware and low-end commercial products do not have a way to specify "Back up everything except what is on the exclude list." If you use one of these products, you will have to go to some effort to find out what "everything" is. You may have to look in the fstab/vstab file in UNIX, or the registry in Windows, to determine the mounted filesystems. UNIX databases often keep a list (e.g., Oracle's oratab file) of what databases live on that server. Windows databases often store such data in the registry. A savvy system administrator could build a script that would automatically determine the names of filesystems, drives, and databases that need to be backed up, and send that list to the backup software product. An example of such a script is included at the end of this booklet.

When to Back Up

Many backup programs will not read a file another process has open. Therefore, the time most often chosen for backups is whenever the files are least likely to be accessed and the network has the least traffic—usually, from 8 p.m. to 8 a.m. With the growth of the amount of data online, it has become increasingly necessary to be somewhat selective about how much data to back up each night. With the emergence of globalization and the existence of 7x24 operations, iquiescent times are shrinking or disappearing altogether. Advanced technologies—mirroring and snapshots—must be called into play, as well as a thorough understanding of system usage patterns and the amount of change from day to day. (The terms above, as well as backup levels, will be covered more thoroughly in the Definitions and Concepts section.)

With larger and larger datasets, a full backup every night becomes an unwieldy goal. For many systems, the amount of data that changes per day can be substantial, resulting in very large incremental backups as well. You must strike a balance between how long it takes to back up data and how long it will take to recover from a disaster. If you did a full backup every night, you would only have to use last night's volume to recover everything. If much of the data doesn't change very often, though, you will be wasting large amounts of media. (At $50–$200 for a single high-capacity tape cartridge, the costs mount quickly.) If you are very frugal and only perform a full backup once a month, with incremental backups the rest of the time, you will have to invest a good deal of time recovering a system.

Deciding How to Back Up

The answer to the "how" question will be influenced by several factors. Most of us work in companies where the bottom line is very important. Thus the financial cost of building the infrastructure to perform backups will influence how they are performed. The size and capabilities of the backup system must be proportional to the data it is backing up. Backing up terabytes onto 4mm DDS-1 cartridges (of 2.4GB capacity) obviously makes little sense. Neither does backing up a 2GB desktop workstation onto a 330GB DST cartridge. How about trying to retrieve many gigabytes of data over a 10baseT network, where 10mbps at best would yield approximately 3.6GB per hour and would fully saturate the network?

There are several approaches to these problems, and new technologies are providing more choices all of the time. Let's look at a few.

Automated Media Libraries

The field of automatic backup hardware is playing catch-up to the capacities of storage devices being marketed today. The sizes of disk drives continue to grow and their cost continues to shrink. Of course, with RAID, JBOD, storage arrays, and SANs, the amount of data needing backup is staggering. Tape media store anywhere from a few gigabytes to several hundred gigabytes per volume. The increasingly common need for terabyte of data storage has spurred the growth of the automated media library industry. These devices can be found with capacities ranging from just a few pieces of media to several thousands. Many can hold multiple media devices as well as multiple pieces of media. Most also have a method for tracking each piece of media using bar-code labeling.

Media, or volumes, are available in several technologies. Many drives have the ability to compress and uncompress the data stream during write/read operations. For comparison's sake, try to find the uncompressed values for a drive, which will let a wise system administrator know the real capacity of the drive. Helical-scan recording devices pass the tape over read/write heads at an angle (actually, the head is at an angle to the tape). This technique allows a very tight packing of bits per linear inch on the tape. Linear drives write data parallel to the edge of the tape in "lanes" from beginning to end, then end to beginning, and so on.

Table 1 compares the tape technologies that are available on the market as of this writing.

Table 1. Tape Technologies

Drive Type	Media Type	Capacity Native/ Compressed	Transfer Speed Native/ Compressed	Load/Unload Time	Access Time	Manufacturers
3570	MP	5/15	2.2/6.6			IBM
8mm (8500)	8mm	5/10	500KB/1MB			Exabyte
9840	9840	20/40	10/20		18	StorageTek
9840B	9840	20/40	19/38		18	StorageTek
9940	9940	60/120	10/20		63	StorageTek
ADR 50	ADR50	25/50	2/4			Onstream
AIT-3	8mm(AIT)	100/260	12/30	10	27	Sony
CD	CD	600MB	300KB W 900KB R			HP
DDS-4	DDS-4	20/40	1. 5/3		50	HP, Sony, Seagate
DLT 8000	DLT	40/80	6/12	12	60	Quantum, Tandberg
DLT1	DLT	40/80	3/6		68	Quantum, Benchmark
DST 314	DCR	100–660	20		18–122	Ampex
DTF-2	DTF	60–200/ 180–600	24/62		21–71 (1.4GB/s)	Sony
Eliant (8mm–8700)	8mm	7/14	500KB/1MB			Exabyte
JAZ 2GB	Jaz cartridge	2	8.7			IOMEGA
LTO Ultrium	Ultrium	100/200	15/30		76–115	IBM, HP, Seagate
MO 9100MB	MO	9.1	6.1 R 3.1 W			HP, Sony
M2 (Mammoth 2)	8mm AME	75/150	15/30	17	60	Exabyte
Orb	Orb	5.7	17.35			Castlewood
SD-3	3490E	10–50	11			StorageTek
SDLT 220 (Super DLT)	SDLT	110/220	11/22	12	70	Quantum, Tandberg
SLR 100	SLR,	50/100	5/10	30	58	Tandberg
Travan NS20	Travan	10/20	1/2			Seagate
VXA-1	VXA	33/66	3/6			Ecrix
ZIP 250 (new!)	Zip cartridge	250MB	1			IOMEGA

Gigabit Ethernet

An important consideration for backups of more than one host is the method by which data travels between hosts. It is often prohibitively expensive to attach a backup device to each host. Therefore we rely on network connectivity to provide the channel for the data. This connectivity is either from Ethernet or, increasingly, SANs, as covered in the next section.

Networking exists today in many forms, but the most widespread implementation is IEEE 802.3 Ethernet. This can exist on many different physical layers, from coaxial cable, to twisted-pair copper, to fibre-optic channels. The speeds range from 56Kb/sec to 1000Mb/sec. (As of this writing, 10 Gigabit Ethernet is just around the corner, offering 10,000Mb/sec.) Obviously, as the amount of the data to be stored is increased, the capacity of the channel needs to match. Even at 1000Mb/sec, it will take at least 8 sec. to transmit a gigabyte of information. The transmission of a terabyte of data, then, is even more intimidating. Since networks are generally a shared resource, the ability to transmit data is constrained not only by the amount of data, but by the number of hosts, non-backup network usage, topology, and equipment. However, a dedicated Gigabit Ethernet network would be more than enough bandwidth to back up many environments, especially if you use hardware-accelerated NICs. (A hardware-accellerated NIC offloads the TCP/IP processing from the host CPU and performs it on the card.)

SANs

Storage Area Networks (SANs) are increasingly popular in today's large data centers. A SAN is essentially a separate network over which all storage traffic runs. Generally, this is not an Ethernet or IP-based network but relies on a separate protocol (Fibre Channel) that communicates data more efficiently. Storage devices (disk drives, tape drives, robot libraries, etc.) are placed directly on the network, and all attached hosts can share the device. Parallel SCSI devices that do not support Fibre Channel may be attached via a SAN router which is then attached to the SAN. Many newer SCSI devices and SAN routers have a SCSI-3 command capability known as *extended copy*. This allows the initiating host to request direct SCSI-to-SCSI copy without having the data travel through the host's SAN interface and CPU.

There are three different topologies to choose from: point-to-point, fibre channel arbitrated loop (FC-AL), and switched fabric. A point-to-point SAN exists between a single Fibre Channel device and a host. In arbitrated loop SANs, only one device can speak at any given time, so they use an arbitration algorithm to determine which device has the right to use the network. On the other hand, all devices on a fabric SAN can communicate simultaneously. Fabrics are built using switches. FC-AL SANs are built using special cabling or Fibre Channel hubs. (SANs will be covered in more detail later in this booklet.)

Concepts

The following section explains some important concepts that are often used when discussing backup and recovery.

Dump Levels

Dump levels are used in many software products ,as well as with native utilities such as (ufs)dump and NTBACKUP. A level specification allows you to selectively back up volatile files without making many duplicates of static files. The specification of a level number directs the backup program or backup software to back up all files that have changed since the last dump at a lower level. A level 0 signifies a full backup (i.e., all files are backed up). A level 2 will back up any files changed since the last level 1 or, if no level 1 occurred, the last level 0 backup. There are important variations on level terminology, so you must read the definition closely to determine the effect of specifying a level. For example, some software allows backup of new or modified files since the *same* or lower level. This means that a week of level 9 dumps will back up the changes that happen each day.

Grandfather, Father, Son

This scheme uses levels to minimize the number of tapes required to recover a directory or filesystem to a particular state. It is based on making a level 0 backup at the beginning (or end) of every month to vault. Every week, a level 0 is taken at the beginning (or end) of the week. Every other day of the week, an incremental backup occurs (either 1,2,3,4,5 or 9,9,9,9,9, depending on the method). The recovery phase would take the most recent full backup and then use the incremental tapes to bring the filesystem forward to the correct point in time. The monthly level 0 is called the *grandfather*, the weekly fulls are the *father* tapes, and the daily incremental backups are the *sons*.

For example, to restore to the 3rd Thursday of a month, where the father tapes are made on Sunday, you would start with the backups made on the Sunday prior to the 3rd Thursday. Then, using the tapes from Monday, Tuesday, Wednesday, and Thursday, the restore process would successively overwrite any changed files and add in any new files that were created during the week. As you can see, this involves a considerable amount of tape handling (though with the proper software and a tape library, the work becomes fairly trivial).

Towers of Hanoi and Enhanced Towers of Hanoi

Better schema minimize the number of volumes required to restore, yet keep files backed up on multiple volumes. One of these schema is based on an old mathematical game called Towers of Hanoi. This game involves moving three rings of increasing size from one peg to the third peg of a three-peg board. Only one ring can be moved at a time, and no larger ring can be placed on a smaller ring. A good URL that talks about the game is *http://www.math.toronto.edu/mathnet/games/towers.html*. The relation of the

moves of TOH to the dump levels is not critical. The point is that two series of numbers are interleaved to create a series which minimizes the number of tapes to restore from a full backup and assures that most changed files are backed up on at least two tapes. Weekly and monthly schedules are shown in Tables 2 and 3

Table 2. One-Dimensional Towers of Hanoi Schedule, Weekly

Sun.	Mon.	Tues.	Wed.	Thurs.	Fri.	Sat.
0	3	2	5	4	7	6

Table 3. One-Dimensional Towers of Hanoi Schedule, Monthly

Sun.	Mon.	Tues.	Wed.	Thurs.	Fri.	Sat.
0	3	2	5	4	7	6
1	3	2	5	4	7	6
1	3	2	5	4	7	6
1	3	2	5	4	7	6
1	3	2	5	4	7	6

In this case, the level 1 backups will capture all files changed since the level 0 on the first Sunday. Thus to restore to the 3rd Thursday, restore from the level 0, the 2nd level 1, the 3rd level 2 and the 3rd level 4.

Table 4. Two-Dimensional Enhanced Towers of Hanoi Schedule, Monthly

Sun.	Mon.	Tues.	Wed.	Thurs.	Fri.	Sat.
0	6	5	8	7	9	8
3	6	5	8	7	9	8
2	6	5	8	7	9	8
4	6	5	8	7	9	8
3	6	5	8	7	9	8

An intriguing alternative is to run a second dimension of incremental levels, as shown in Table 4, where the second dimension handles the Sunday backups and the first dimension covers the weekdays. The number of tapes per restore is the same. You will note that in the monthly schedule (Table 3), the Sunday level 1 backups will con-

tinually increase in size, since they each incorporates all the changes since the beginning of the month. The ETOH (Extended Towers of Hanoi) Sunday backup size is limited to, at most, two weeks' worth of changes. (See *http://etoh.wopr.net/ex.abstract.html* by Vincent Cordrey and Jordan Schwartz.)

Volume Storage Management

The importance of physical security for the backup media cannot be underestimated. If those volumes are missing or destroyed, you might as well never have backed up the files. It is not uncommon to find backup volumes piled, stacked, and randomly distributed throughout a computer room. Disorder like this virtually guarantees that a volume will be lost or misplaced. Do your best to keep your tapes well organized.

Onsite Storage

There are many products available for media storage. Some have drawers sized to hold thousands of tapes. Some are fireproof. Remember, though, that fireproof does not mean heatproof. When a cabinet full of tapes stands near a significant heat source, the tapes may melt or the tape cartridge may warp.

You will also probably want to limit user-level access to the media. Physical access to the media opens the door to misuse of the backup tapes, which may contain confidential information. It also, of course, significantly increases the chances that a particular piece of media could become damaged.

You should keep a tape inventory detailing the location and usage of each piece of media, so that you can find the right volume when needed. Use your up-to-date inventory to perform media inspections regularly.

Offsite Storage

Offsite storage can be accomplished in many different ways. You can take your tapes home. You can store them in a nearby building. You can pay a storage vendor to store them for you. All of the considerations for onsite storage also apply to offsite storage. Above and beyond those considerations, you must keep track of how the offsite vendor stores your tapes. Minimally, if you have good communication and tracking methods, you will be able to specify to the vendor that you need volume A00533 from rack 12, shelf 2, slot 22. Ideally, you'd like to give just the volume label, A00533.

Test their security. Will they leave you alone in their vault to perform an inventory? If so, that means that other customers could be left alone to snoop, steal, or otherwise access your company's confidential backup volumes. Make spot checks and ask for (accompanied) inspections.

Mirrored Root Disk

Many UNIX and Windows operating systems now come with volume management software that allows you to mirror disks onto additional drives. Sun uses Online Disk Suite (ODS) and HPUX has Logical Volume Manager (LVM), for example. If the original root disk fails, you can continue running with the mirrored copy of the operating

system. Note that this will work only for disk failures. File corruption will of course be copied to the mirror, corrupting it too!

Standby Root Disk

With UNIX, you can keep an offline disk available to boot from by manually (using cron) synchronizing the original with the spare and then rebooting from the spare when needed. Of course, if there is file corruption (security breach, bad patch, etc.) on the original, it will be transmitted to the spare and the spare will be just as bad. But by delaying the copy of data from the primary to the standby disk, you can test new patches and versions of the operating system on the primary disk while keeping the old version ready to go on the standby disk. Both of these solutions (mirrored and standby) require an extra disk and extra software, and they are vulnerable to propagation of corruption.

Bare Metal Recovery

A bare metal recovery means recovering an entire system from scratch. This can be required because a disaster has occurred (say, the host was destroyed by lightning) or it can be a planned upgrade or replacement. You will have to restore the operating system, as well as the data disks and partitions. If you must rebuild a system from scratch, it will not be as easy as an operating system reinstall from the original CD. You will lose all patches, modifications, host-specific information, etc. Thus, it works best to use an image of the system disk.

There are many paths to choose when deciding on how to create and recover from this scenario. There is no native bare metal recovery tool for Windows, but there are many Windows-based bare metal recovery products. There is also a free bare metal recovery tool called Mondo Rescue that works for both Windows and Linux on Intel platforms. You can find out more about Mondo Rescue at *http://www.microwerks.net/~hugo/ or http://lists.sourceforge.net/lists/listinfo/mondo-devel/*.

Native bare metal recovery tools are available for AIX, HP-UX, Irix, Tru64, and VMS. Other versions of UNIX will require a third-party tool. A few are available for Solaris. (Please don't ask why this popular version of UNIX does not have its own bare metal tool.)

It is possible to roll your own tool. The following summarizes the steps required for a generic bare metal recovery plan. Please consult W. Curtis Preston's *UNIX Backup & Recovery* for the details for your version of UNIX.

- Back up the operating system using a native backup tool, such as dump.
- Save the partitioning information for your root disk somewhere you can access it in case of failure.
- Recover the boot system into single-user mode, either over the network with a boot server or from CD-ROM.
- Using the partitioning information saved above, set up a new root disk to look the same as the old disk and mount it.

- Recover the OS to the mounted disk.
- Place a boot block on the mounted disk (e.g., Solaris uses installboot).
- Reboot.

2. Survey of Software

There are many, many software choices for backup and recovery. They range from native utilities such as (ufs)dump and NTBACKUP, to enterprise-level commercial solutions. In this chapter we will focus on native utilities and scripts for small, controlled environments. After a brief description of these native utilities, we will cover two popular freeware programs, AMANDA and hostdump.sh.

Native Utilities

These commands are available in most versions of UNIX or Windows. As with any command, the manual or help page is the first place to look for syntax and usage.

tar

The Tape Archive Utility is included with virtually every flavor of UNIX in existence today, and it is also available for Windows. The freeware version from GNU, gnutar, expands the functionality somewhat. For example, gnutar will preserve the access times of files across creation and extraction from a tar archive. The basic syntax for tar is:

```
$ tar [cx]vf device pattern
```

The options are create or extract, verbose, device filename. Device is the archive to create or extract from while the pattern is the files to add (extract) to (from) the archive.

Examples:

Create an archive of the current working directory:

```
$ tar cvf currdir.tar
```

Create a tape archive, rewinding the tape upon completion, of /etc:

```
$ tar cvf /dev/rmt/0 /etc
```

Tar can be used in pipes to great effect to move groups of files around in the system across mount points and NFS mounts, and even between hosts with automounting.

Here is an example of recreating /usr/local from one host to another:

```
$ cd /usr/local
$ tar cvf - . | (cd newhost:/usr/local; tar xvf -)
```

Of course the verbose option is optional to obviate the listing of all the files as they are added, then extracted from, the STDOUT/STDIN archive name (-).

One advantage of tar files is that they can be read with most standard Windows unzipping programs such as WinZip.

cpio

A powerful utility, cpio (**C**o**P**y **I**n/**O**ut) requires careful construction on the command line to accomplish what tar (mentioned above) and dump (mentioned below) can achieve with less verbosity. However, cpio can incorporate find-like functionality to limit the files backed up.

The basic command structure for a backup is:

```
$ find . –print | cpio –o aBcv > device
```

and for a restore:

```
$ cpio –I [Bcv] patterns < device
```

dd

A more basic utility in the UNIX toolkit is dd. Though simple-minded, this program can provide backup and recovery functions but is more useful in retrieving images from partially damaged media. The basic format of the command is:

```
$ dd if=device of=device bs=blocksize
```

The input and output devices are either physical devices such as tapes (/dev/rmt/0c) or files. If the device to be used is STDIN or STDOUT, it need not be specified. The blocksize parameter is the amount of data transferred in one I/O operation, specified in bytes, kilobytes (number followed by a k suffix), or megabytes (number followed by an m suffix). This option is generally used with tape operations to take into account the gap between records: a tape written at a particular block size must be read with the same block size or integer multiples of that size.

A very desirable feature of dd is the ability to do data conversions on the fly: mapping from uppercase to lowercase, translating ASCII to EBCDIC, swapping byte order, etc.

dump/restore

Native to all modern flavors of UNIX are the dump and restore utilities. On Solaris, they are known as ufsdump and ufsrestore due to their ability to handle the UNIX (equivalent to Berkeley Fast) Filesystem. These tools form the basis for many homegrown scripts for backup and recovery and are the general workhorse native utilities for making backup tapes and recovering files on many small and intermediate sites. For full details on these commands, use your local man pages. Table 5 shows the dump commands for various UNIX systems.

Table 5. Dump Commands

HPUX	Solaris	SCO	Network Appliance	AIX	Linux	SGI	DGUX
(r)dump	Ufsdump	xdump	dump	backup	dump	Dump	dump and vdum

Usage:

```
$ ufsdump – [level][update flag][destination flag] arguments
directory-to-backup
```

For example:

```
$ ufsdump –0uf /dev/rmt/0 /home
```

This will back up /home to the tape, /dev/rmt/0, and update the /etc/dumpdates file. Of course, the destination file or device can be STDOUT (and ufsrestore can read from STDIN), making it easy to create pipes:

```
$ ufsdump –0uf - /home | (cd /newhome; ufsrestore –rf -)
```

The –r in the restore command says to restore the entire dumped volume, *home*, to the destination, *newhome*.

Please note that the usage of dump varies from operating system to operating system; care must be taken to adapt to the local conditions. This is especially important to bear in mind when using scripts that rely on the native commands to perform backup and restore.

NTBACKUP/BACKUP

NTBACKUP is the native backup command for Windows NT and 2000. The tool underwent a significant rewrite with Windows 2000 and is now maintained by VERITAS Software, the makers of Backup Exec and NetBackup. The following description of this tool's functionality is taken from its useful help page.

Using Backup, you can:

- Back up selected files and folders on your hard disk.
- Restore the backed-up files and folders to your hard disk or any other disk you can access.
- Create an Emergency Repair Disk (ERD), which will help you repair system files in the event they get corrupted or are accidentally erased.
- Make a copy of any Remote Storage data and any data stored in mounted drives.
- Make a copy of your computer's System State, which includes such things as the registry, the boot files, and the system files.
- Back up services on servers and domain controllers, including such things as the Active Directory directory service database, the Certificate Services database, and the File Replication service SYSVOL directory.
- Schedule regular backups to keep your backed up data up to date.

You can use Backup to back up and restore data on either FAT or NTFS volumes. However, if you have backed up data from an NTFS volume used in Windows 2000, it is recommended that you restore the data to an NTFS volume used in Windows 2000, or you could lose data as well as some file and folder features. For example, permissions, encrypting filesystem (EFS) settings, disk quota information, mounted drive information, and Remote Storage information will be lost if you back up data from an NTFS volume used in Windows 2000 and then restore it to a FAT volume or an NTFS volume used in Windows NT 4.0.

Free Software

Very good, freely available backup packages include the Advanced Maryland Automated Network Disk Archiver (AMANDA) and hostdump.sh. This is by no means an exhaustive list. We'll cover AMANDA in depth because it is a mature product, free, and relatively powerful compared to home-grown scripts.

AMANDA

AMANDA, developed at the University of Maryland, is probably the most widely installed free backup utility, with an install base of over 1500 sites. This utility is available from http://www.amanda.org. Written primarily by James da Silva of the Department of Computer Science at UM, AMANDA allows you to set up a single master backup server to backup up multiple hosts to a single backup drive or tape robot. AMANDA uses native dump and/or gnutar utilities and can back up a large number of workstations running multiple versions of UNIX. It was one of the first freeware utilities to address backups of multiple hosts onto a central server.

With the introduction of high-capacity tape drives in the early 1990s and the size of the largest disk drives at the time (1–2GB), the space on a tape exceeded the storage size of a typical workstation. This naturally led to the practice of backing up multiple hosts onto a single tape. Coordinating access and providing tape hardware became prohibitive in effort and cost. Furthermore, gathering the data streams from multiple hosts over the network to the high-speed tape drive often encountered the bottleneck of network bandwidth.

AMANDA's solution to this situation is to use a holding disk to gather the data on the tape server host before streaming the blocks onto the tape. An independent process drains the data from the holding disk onto the tape device. This allows a holding area to be kept to a moderate size, since it is being emptied faster than it is being filled.

Scheduling differs from the traditional grandfather-father-son paradigm. A dump cycle is defined for each area to control the maximum time between full dumps. AMANDA looks at that information and at statistics of past dump performance in order to estimate the size of dumps for this run and decide which backup level to do. This allows the software to balance the dumps per day such that the runtime is roughly constant from day to day. The effect is that AMANDA will attempt to perform *n* level 0 dumps of the specified filesystem in each dump cycle, but it is not guaranteed that those dumps will fall on any particular day.

AMANDA supports multiple configurations on the same tape server. This allows preset schedules for periodic backups (such as quarterly full backups), as well as normal daily configurations. Multiple configurations can run in parallel if multiple tape drives are available.

The configuration of AMANDA determines how effective it is. After proper configuration, logs will show the progress, success, or failure of any particular job. Because this utility is run from a cron job, the invocation is less important than the configura-

tion, so the following paragraphs will deal with configuration instead of the command line invocation. A cohesive, in-depth description can be found in W. Curtis Preston's *UNIX Backup and Recovery*. In addition, the AMANDA home page, *http://www.amanda.org*, offers links to AMANDA discussion mailing lists.

Tape Server Configuration

The tape server acts as the master controlling host for all AMANDA operations. AMANDA can be CPU-intensive if it is configured to do server-side compression, and it is almost always network- and I/O-intensive. It typically does not use much real memory. It needs direct access to a tape device that supports media with enough capacity to handle the expected load.

To get a rough idea of backup sizes, take total disk usage (not capacity), USAGE, and divide it by how frequently full dumps will be done, RUNS. Pick an estimated run-to-run change rate (how much the filesystems are changing), CHANGE. Each AMANDA run, on average, does a full dump of USAGE/RUNS. Another USAGES/RUNS*CHANGE is done of areas that got a full dump the previous run. Further, USAGE/RUNS*CHANGE*2 is done of areas that got a full backup two runs ago, etc.

For example, with 10 in use, a full dump every seven runs, and estimated run-to-run changes of 5%:

100GB / 7	=	14.3GB
100GB / 7 * 5%	=	00.7GB
100GB / 7 * 5% * 2	=	1.4GB
100GB / 7 * 5% * 3	=	2.1GB
100GB / 7 * 5% * 4	=	2.9GB
100GB / 7 * 5% * 5	=	3.6GB
100GB / 7 * 5% * 6	=	4.3GB
	=	29.3GB

If 50% compression is expected, the tape capacity—which can be spread over multiple tapes—needed for each run would be 14.7GB. This estimate could be improved with greater knowledge of actual usage, but it should be close enough to start with. It will also yield an estimate of the time required for each run: just divide the capacity by the tape speed.

The tape device specified must be a non-rewinding type, and it is highly recommended that the tape drive have integrated hardware compression on the drive. The non-rewinding drive will leave the tape positioned ready to receive the next stream of data after a particular filesystem is backed up without having to wait for the rewind and forward spacing to locate the current end of data position on the tape. Compression on the drive will alleviate the tape host from the CPU intensive task of compression. Compression only makes sense to implement on clients with low bandwidth connections to the tape host. This will minimize the number of bytes that have to travel over that connection to be staged to the holding disk.

amanda.conf

The main configuration file—usually */etc/amanda/configurationname/amanda.conf*—contains many tunable parameters which are documented within a sample file in the distribution. However, some of the concepts could do with a little additional explanation. Here are a few of the more intricate variables to tune and a discussion of their usage:

dumpcycle: How often to perform full dumps. Short periods make restores easier because there are fewer incrementals, but they use more tape. Longer periods let AMANDA spread the load better but may require more steps during a restore. The amount of data and the capacity of your tape drives also affect the dump cycle. Choose a period long enough that AMANDA can do a full dump of every area during the dump cycle and still have room in each run for the partials. Typical dump cycles are one or two weeks. Remember that the dump cycle is an upper limit on how often full dumps are done, not a strict value. AMANDA runs them at different frequencies and at various times during the cycle as it balances the backup load.

This is both a general amanda.conf parameter and a specific parameter set for each dumptype. The value specified in a dumptype takes precedence over the general parameter. To handle areas that change significantly between runs and should get a full dump each time, such as a mail spool or system logs, create a dumptype based on another dumptype, changing attributes as desired (e.g., client dump program, compression), and set dumpcycle in the new dumptype to 0:

```
define log-dump {
 comp-user-tar
 dumpcycle 0
}
```

To run full dumps by hand, i.e., outside of AMANDA (perhaps they are too large for the normal tape capacity or need special processing), create a new dumptype and set strategy to *incronly*:

```
define full-too-big (
 comp-user-tar
 strategy incronly
}
```

dumptype: A set of specifications that supersede the global settings for a particular class of backups. In the two examples above, comp-user-tar is a preexisting dumptype upon which a further dumptype is specified. The comp-user-tar type will compress backups of user directories and uses the tar program, rather than a native dump utility, as the basis of the backup.

runspercycle: The number of times during a cycle that a filesystem or set of files will be backed up. For example, if the cycle is 7 days and runspercycle is 5, the backups are performed 5 days out of 7 (i.e., weekdays).

runtapes: Normally, AMANDA uses one tape per run. With a tape changer, the number of tapes per run may be set higher for extra capacity. AMANDA uses only as much tape as it needs, but it does not yet do overflow from one tape to another. If it

hits EOT while writing an image, that tape is unmounted, the next one is loaded, and the image starts over from the beginning.

tapecycle: How many tapes will be used per cycle. This is either explicit or implied from the runtapes and runspercycle. To ensure that the current run is not overwriting the last full dump, one more run should be included. For instance, a dump cycle of two weeks, with default runs per cycle of 14 (i.e., every day) and default tapes per run of 1, needs at least 15 tapes (14+1 runs times 1 tape/run). AMANDA allows backup tapes to be reused.

Holding Disk

The holding disk is one of those great ideas few other products use. It acts as a buffer to maximize tape throughput when the data is being transported over slow media such as a network.

Define each holding disk in an amanda.conf holdingdisk section. If you are dedicating partitions to AMANDA, set the use value to a small negative number, such as –10MB. This tells AMANDA to use all but that amount of space. If space is shared with other applications, set the value to the amount AMANDA may use, create the directory, and set the permissions to allow only the AMANDA user to access it.

Set a chunksize value for each holding disk. Negative numbers cause AMANDA to write dumps larger than the absolute value directly to tape, bypassing the holding disk. Positive numbers split dumps in the holding disk into chunks no larger than the chunksize value. Even though the images are split in the holding disk, they are written to tape as a single image. At the moment, all chunks for a given image go to the same holding disk.

Older operating systems that do not support individual files larger than 2GB need a chunk size just slightly smaller, say 2000MB, so that the holding disk can be used for large dump images. Systems that support individual files larger than 2GB should be assigned a very large chunk size, such as 2000GB.

Configure Clients

On the AMANDA tape server, after tapes have been labeled according to the labelstr specifications, pick the first client, often the tape server host itself, and the filesystems or directories to back up. For each area to back up, choose either the vendor dump program or gnutar. Vendor dump programs tend to be more efficient and do not disturb files being dumped but usually are not portable between different operating systems. The free utility gnutar is portable and has some additional features, such as the ability to exclude patterns of files, but it alters the last access time for every file backed up and may not be as efficient as the vendor-supplied utility. However, gnutar may deal with active filesystems better than vendor dump programs, and it is able to handle very large filesystems by breaking them up by subdirectory.

Choose the type of compression for each area, if any. Consider turning off compression of critical areas needed to bring a machine back from the dead, in case the decom-

pression program is not available when you need to do the restore. Client compression spreads the load over multiple machines and reduces network traffic, but it may not be appropriate for slow or busy clients. Server compression increases the load on the tape server machine—possibly by a factor of two or even more, since multiple dumps are done concurrently.

Pick or alter an existing dumptype in the amanda.conf file that matches the desired options, or create a new one. Each dumptype should reference the global dumptype, which is used to set options for all other dumptypes.

Create a file named disklist in the same directory as the amanda.conf file and either copy the file from the example/disklist or start a new one. Make sure it is readable by the AMANDA user. Each line in the disklist defines an area to be backed up. The first field is the hostname (FQDN recommended), the second field is the area to be backed up on the client, and the third is the dumptype. The area may be entered as a disk name (sd0a), a device name (/dev/dsk/c0t0d0s0), or a logical name (/usr).

Recovery

AMANDA restores files easily with the amrecover command. However, to get the full power of amrecover, you must tell AMANDA to retain file indices on its runs. This is configured through the global dumptype index parameter. This must be set to yes and the amindexd and amidxtaped services must be installed and enable for inetd on the tape server machine. The amrecover will run as a root process on the client host to recover the specified file.

Amrecover starts an interactive program that allows you to browse the files to recover. Amrecover finds the tapes which contain the images, prompts you through mounting them in the proper order, searches the tape for the image, brings it across the network, and pipes it to the appropriate recovery utility on the client.

For full filesystem recovery, the amrestore command is used, such as:

```
# amrestore –p amanda-tape-name myhostname filesystemname |
ufsrestore –rf –
```

This will pipe the tape stream into the ufsrestore program.

hostdump.sh

Hostdump.sh is a plug-and-play script to make a backup on a given host quickly. It supports up to 20 versions of UNIX. It is available from *http://www.storagemountain.com/hostdump.html*.

To use it, simply make sure there's a volume in the tape drive, then invoke hostdump.sh with a device name and one or more host names, and it will back up all the hosts that you list to the device you specify. It automatically determines the names of all the filesystems, as well as their filesystem types. If it is a filesystem type that supports dump, it calls the appropriate command. If it is an unknown type or a UNIX system that does not have a good dump command, then utility will use the cpio command. It puts two extra tar files on the tape. The first file is a header that lists all the filesystems on the volume and the commands that were used to back them up. After all

backups are done, it then rereads the table of contents of each of the backups and places that information into a second tar file at the end of the volume. Detailed instructions on how to read the tar file at the end of the volume are also in the header file on the first partition.

Hostdump Usage

```
# hostdump.sh level device logfilesystem [system2 … systemx]
```

To back up less than the entire system, after the system name add the name of the filesystem you want. If you want to back up more than one filesystem from the same system, you will need to specify the system name before each filesystem. Issue the following command:

```
# hostdump.sh level device logfilesystem:/filesys system:/
  filesys
```

where

level: A valid dump level between 0 and 9.

device: A non-rewinding tape device, such as /dev/rmt/0n, or /dev/nrmt0. (The script will test to see whether the device you specify is actually a no-rewind device.)

logfile: An absolute pathname to a logfile that will contain STDOUT and STDERR from script. This pathname should be a name that can be associated with the tape. You could use something like this:

```
/backuplogs/apollo.DevRmt0n.Level0.12.13.1996
```

system1 [system2 … systemx]: A list of one or more systems that you want the script to back up to the device you listed with the second argument. Every system you list will be written to the tape, in the order they were listed. The script will automatically look at each system's fstab file and create a list of filesystems to be backed up.

system:/filesys: [system:/filesys]: A list of which filesystems to back up. (If you want to back up the whole system, use the preceding command and let hostdump.sh figure it out for you.)

system1 system2 system3:/filesys system4:filesys: You can also mix and match the options like this. Any systems that are listed without an accompanying filesystem name will receive a full backup. Any systems that have a filesystem listed after them will have only that filesystem backed up.

Advanced Option 1: Special Case Filesystems

You may want to exclude some filesystems in the fstab file on a regular basis. Or you may want to back up the /tmp filesystem, which is normally excluded by hostdump.sh. hostdump.sh can handle both of these special cases. If you want to exclude filesystems that would normally be included, you put that filesystem's name in a file called fstab.exclude on the system from which you want to exclude it. (For "fstab" you need to substitute whatever your version of UNIX calls the fstab file. For example, you would use /etc/vfstab.exclude on Solaris.) To include filesystems that are not in the

fstab file or are normally excluded by hostdump.sh, put that filesystem's name in a file called fstab.include on the system in which you want to include it.

For example, assume that on a Solaris system called apollo, you want to exclude the /home filesystem. Normally it would get backed up, since it is in /etc/vfstab. Suppose you do want to include the /tmp filesystem. You would create two files on apollo.

One would be called /etc/vfstab.exclude, and it would contain the entry:

```
/home
```

The second file would be called /etc/vfstab.include, and it would contain the entry:

```
/tmp
```

Advanced Option 2: Systems Bigger Than a Tape

hostdump.sh, like the dump and restore commands, was not originally designed with today's systems in mind. The script was first written to back up Ultrix systems, the largest of which was 7GB, to 8mm compressed drives (the smallest of which was 10GB). The original author never envisioned a system where the disk was bigger than a tape. Then hostdump.sh met HPs that shipped with 20GB of disk and one 6GB DDS tape drive! Something had to be done.

This is where backup philosophy enters again. Please remember the essential elements of a good backup:

- Automation (should not require you to swap tapes in the middle of the night)
- Intelligence (should figure out for itself what to back up—include lists bad!)
- Comprehensiveness (don't forget anything!)

The easiest way to solve the problem would have been to scrap the original idea of looking at the fstab and just give the utility a list of filesystems to back up. Essentially, that's what's been done, but with a twist. Try this:

- Use hostdump.sh in the regular manner, excluding enough filesystems in fstab.exclude so that what is left will fit on one tape.
- Use the advanced option to include the excluded filesystems onto another tape or tapes.

Why is it important to use this method? It goes back to the second essential element of a good backup: intelligence. Say, for example, that you have two include lists. You tell hostdump.sh to back up /, /usr, /var, /opt, and /home1 on one tape, and /home2 and /home3 on another. What happens when you add /home4? Unless someone or something tells the backup program about it, it will never get backed up. You'll never know there's a problem, either, until someone asks you to restore /home4/yourboss/really-important-presentation-to-the-board-of-directors.doc. So, whether you are using hostdump.sh or not, you need to find a way to back up the entire system onto one tape by excluding certain filesystems, then back up the excluded filesystems onto another tape. That way, when you add /home4, it will automatically be included on the first tape. The worst that could happen would be that the new

/home4 would overflow your first tape (which you will know immediately, because you monitor your backups, right?), and you would have to add it manually to the second tape and exclude it from the first.

To use this option, run hostdump.sh a second or third time, specifying the filesystems that were excluded in fstab.exclude. It is easiest to explain this option with an example. In this example, elvis is the name of the Solaris system you want to back up. You are using hostdump.sh for the first time, and you know that a full backup will fit onto three tapes. You have divided up the filesystems equally in the following manner:

tape 1: OS filesystems (/, /usr, /var, /opt, etc.), /home1

tape 2: /home2, /home3, /home4

tape 3: /home5, /home6

First, back up the whole system, excluding /home2–6, on the first tape by creating a file on elvis called /etc/vfstab.exclude that contains the following lines:

```
/home2
/home3
/home4
/home5
/home6
```

Run this command every night:

```
# hostdump.sh level device1 blocking-factor logfile1 elvis
```

This will back up the entire system, excluding what is in /etc/vfstab.exclude.

Second, back up the other filesystems on other tapes. Run these two commands every night:

```
# hostdump.sh level device2 blocking-factor logfile2 elvis:/
  home1 elvis:/home2 elvis:/home3 elvis:/home4
# hostdump.sh level device3 blocking-factor logfile3 elvis:/
  home5 elvis:/home6
```

Backup Software Summary

Commercial packages have much greater capabilities than the freeware and native utilities. They scale better, have better performance tracking and reporting utilities, and support a broader range of configurations (O/S, tape libraries, network configurations, SANs, etc.). Of course, you pay for what you get. Usually each option requires purchasing an extra-cost license and limits the number of hosts per license (more hosts = more cost). Native utilities, though, exist on every flavor of UNIX and Windows and operate in similar fashion. Freeware utilities bridge the gap between the costly commercial solution and the arcane UNIX and Windows commands. A good survey of the commercial offerings can be found at *http://www.storagemountain.com*.

3. Database Backup & Recovery

Performing regular database backups is one of the hardest tasks on the plate of today's system administrator, because databases tend to be far larger and more complex than other files. In order to properly back up a database, you need to:

- Understand the internal structure of your database.
- Understand the available utilities.
- Have an excellent working relationship between system administrators and database administrators.

Once you've accomplished all of that, you'll need to choose among your various options:

- Buy an expensive commercial utility.
- Find or write your own utility.
- Perform cold backups without a utility.

Almost anyone who reads this list will find at least one of these steps daunting. Many people work with databases that operate 24 hours a day, 7 days a week. They can't shut them down for hours at a time to back them up. Even if they could, if a database uses raw devices it can't be backed up with a regular dump. Of course dd would work, but that would mean doing one thing for filesystems and a different thing for databases. A common theme throughout this book is that Different is Bad. Every special case is a chance for failure. It's something else you have to code for, something else you have to watch, something else that could break. The bottom line: database backups are not easy.

Part of the problem is the design of the database engine itself. Historically, the need for bigger storage and faster queries drove the design of a product much more than its ability to back itself up. (This goes for filesystems too. Most UNIX vendors have added support for a multiple-terabyte filesystem that break the 2GB file-size barrier, but at least one man page for dump says "WARNING: dump will not back up a filesystem containing large files.") Over the last few years, databases have gone from a gigabyte or so to an average size that is daily growing closer to a terabyte. This growth in size and performance happened because the customer base screamed for it. Unfortunately, they weren't simultaneously screaming for a backup utility to support those huge databases.

Can It Be Done?

Think about database backups from a big-picture perspective, comparing them to filesystem backups. There are a number of good backup utilities on the market now. Why aren't there just as many for database backups? The demand certainly exists.

One reason for the paucity is the complexity of the task. In order to release a database backup product, a company would need to take into account several factors:

Multiple moving targets

How do you get a database to hold still? Have you ever tried to take a picture of 100 people? Designing a backup utility for a database is very hard, since you have to "take a picture" of hundreds of files at once.

Interrelationships among the files

A database backup program needs to understand all the database elements and how they relate.

Working with the database engine

This is essential. If the database understands that you are running a backup, it can help you. If you don't interface with the database, you'll be backing up blind.

The size of the job

How do you get one terabyte of data to a backup drive in one hour? That's what some backups require! The only answer is a multi-threaded backup program.

Recoverability versus cost

You need all of the preceding, but you don't want to mortgage your business to get it.

Differing levels of automation

Different customers want different levels of automation. Some want everything managed by the library, and others would rather do it themselves.

With these kinds of requirements, is there any hope of getting commercial utilities that are up to the challenge? The answer is "Yes!" Database companies have finally recognized that backup utilities do have an effect on the overall sales of a database engine. They have finally started producing good utilities that interface with other commercial backup products. Vendors have even taken the lead in making sure that customers use a utility that works properly for both backup and restore. Now that there are decent backup utilities, though, you have another problem—confusion.

What's the Big Deal?

Why are we hearing so much about database backups all of a sudden? Why are they so hard? Why don't utilities currently exist to do all this? Can't I just shut down the database and back up the whole system? These are all questions that may be going

through your head. If you already know the answers, feel free to skip to the next section.

Why are we hearing so much about database backups all of a sudden?

The demand for Relational Database Management Systems (RDBMSes) has grown exponentially in the last few years. Not only are there more databases, they are faster, larger, and more complex than ever before. Companies are relying increasingly on bigger and bigger databases to store their information—information that, if lost, could never be replaced. Customers have started to recognize the importance of safeguarding their data, and the demand for better backup and recovery utilities has followed. Database companies and backup product companies have finally responded with utilities that are up to the task.

Why is backing up a database so hard?

Actually, backing up a database isn't that hard. It's restoring the database that has caused many people to go insane! Seriously, though, the reason it is so difficult is that you need both a good system administrator and a good database administrator in order to design a workable backup plan. Most people know only one side well. If you or your people know both sides, then consider yourself very lucky. In some companies, it's tough to get the two sides to work together.

Why aren't utilities already available to do all this?

Backups aren't sexy. Historically, customers have asked for faster databases or easier-to-program databases. Whether a good backup utility existed was not even considered until well after a product was purchased, installed, and, quite often, in production. Many times it took a disaster to get some people to realize that backup and recovery are essential for any database system. This has finally changed. Maybe customers finally realized that they needed to ask about backup and recovery when they were evaluating a database product. Maybe the database companies' support departments beat up the developers because they were spending all their time on down system calls. (Hell hath no fury like a customer who has suffered data loss that could have been prevented by a better backup utility.)

One of the most important features customers needed was the ability to integrate their database backups into their commercial backup utility. Remember, though, that these products haven't been on the market for long. Until a few years ago, if you asked your database vendor if its database worked with product X, they were likely to say, "What is that and why should it?" It's not clear who broke the barrier first, but all three big vendors did the same thing within a year or two of one another—they cooperated with commercial backup companies to develop and release their own

utility that was designed specifically to work with third-party backup products.

Can't I just shut down the database and back up the whole system?

In a small numbers of cases, yes. However, a number of considerations might prevent you from doing this, including your database platform, whether you are using raw or filesystem files, and whether you need point-in-time recovery. Those details are covered in appropriate chapters in W. Curtis Preston's *UNIX Backup & Recovery*.

What Can Happen to an RDBMS?

A lot can happen to interfere with the normal operation of a database. What you need to do to get the database running again will depend on what broke it. Some problems you may encounter:

Device ownership change

Someone can accidentally change the ownership of the raw devices or files the database is using as datafiles. Since the database can no longer write to the files, it will cease to function. You will need to return the device to its proper ownership and, possibly, restore data.

Device permissions change

This is similar to an ownership change, since the database engine can no longer write to the file. The fix is the same as that for ownership change.

Device symbolic link removed

In UNIX, you are well advised not to use the actual raw device when setting up a database. You should make a symbolic link to a name that makes sense (e.g., ln -s / dev/rdsk/c0t0d0s0 /dev/informix/chunk1). This allows much greater flexibility if that device goes bad. However, you need to document what device the database is pointing to, so that you can remake the link if someone deletes it. (You can also restore this information from backup, but it is much faster just to remake it, if you know what to link it to.)

Disk goes bad

The only real protections against a bad disk are mirroring and backups.

Controller goes bad

If you are mirroring some of your devices, you should set up the mirroring so that a device on one SCSI controller is mirrored to a device on another SCSI controller. This ensures that if a SCSI controller does go bad it won't take out both mirrors at once.

Database raw device assigned as a swap or filesystem

This one's a bad one. Proper documentation helps a lot. It also helps if your administrators are trained to look at the ownership of a device before they

use it. If it is owned by someone other than root, they should never use it. To recover from this error, you need to undo the change and restore from backup.

Another application uses the raw device as a mirror

This is similar to the preceding scenario; an administrator tries to use one of the database disks for something other than the database. Again, you will have to undo the change and restore from backup.

Backing Up an RDBMS

Protecting an RDBMS is very complex. There are several storage elements, including datafiles, rollback logs, transaction logs, and the master database. How do you get all of the data to a secondary storage medium if it's constantly changing?

Physical and Logical Backups

There are two primary methods of backing up an RDBMS: physical backups and logical backups. A *physical*, or *database*, backup physically backs up the data files. Physical backups may be *cold* or *hot*. In a *cold backup*, the database shuts down for the duration of the backup. This is often the simplest method., If your database's data files reside in the filesystem, with the database shut down you can run your normal filesystem backup utility. Unfortunately, this method may require your database to be shut down for a long time. That is why more and more environments are performing hot backups, done while the database is online. This method, of course, takes a lot more work behind the scenes, since you are trying to copy the data files while the database is writing to them. You need a backup utility that understands the internal structure of the database. The purpose of this utility is to log the changes to a particular datafile while it is being copied to the backup media, yielding a consistent backup image.

A *logical backup* copies, or exports, data *objects* (usually tables) but does not record data locations. A logical backup can be used to restore a deleted table without having to restore all of the datafiles in which it resides. It can also be used to move a table from one database to another. Since a logical backup backs up only the data and not its locations, data can be restored into any location. Logical backups, however, do not have the ability to do a point-in-time recovery. Moreover, they can introduce referential integrity problems, since you could load a table that requires information from another table that is not present. The biggest problem with exports, though, is that they almost always need to be done with the database offline.

Physical backups may be performed in any of several ways:

- If you are using Oracle or Sybase, and your datafiles are cooked files, you can simply shut down the database and do a full system backup. Since all the information exists as regular filesystem files, you will get everything backed up. You could restore the entire database from such a backup, so long as you remembered to replay the transaction logs against the old data-

base files. This step is why you cannot do a cold backup with Informix; Informix has no way of replaying the transaction logs without restoring from a backup that was made with its backup utility.

- If you are using Oracle or Sybase, and your datafiles are raw partitions, you still can shut down the database and back them up if you have a utility or script to do so. For example, in UNIX you would use dd. This is quite a bit more complex than the first method, though, because you need to know which devices to run dd against.
- Another method is to back up the database live to disk or tape, using a utility provided for that purpose. Informix provides the ontape utility and Sybase provides the dump utility. Oracle does not have such a utility, but it does let you write your own. (Oracle's alter databasebegin/end backup commands allow you to back up an Oracle database in a number of ways.) This provides a lot of flexibility. If you're not really good at scripting, you can use the public-domain utility oraback.sh, available at http://www.storagemountain.com.
- The newest method of backing up databases is to use a utility that sends one or more streams of data to a commercial storage manager (i.e., backup software). This is the cleanest method if you can afford it (it costs several thousand dollars per system). Each of the three major database vendors provides such a utility: Informix provides Online Backup and Recovery (onbar), Sybase provides dump, Oracle7 provides the Enterprise Backup Utility (EBU), and Oracle8 provides Recovery Manager (rman).[1]
- A few commercial tools provide yet another kind of functionality. Wrapping around some of the native utilities, they let you send a data stream to commercial backup products, some of which offer interfaces to these utilities. The most popular of these utilities is SQL Backtrack, which is now sold by BMC Software. Still other options include interfaces to products that do not use the newer interfaces. For example, many vendors prefer not to use Oracle's EBU and have written commercial interfaces that use the same native commands as oraback.sh. The vendors claim more reliability and/or faster performance. The validity of backup and recovery programs that do not use the vendor-supplied API is left as a decision for the reader.

Performing a logical backup is actually much simpler than doing a physical backup. Each of the databases comes with an export utility that creates a logical backup of one or more database objects to a file. Some of the commercial utilities allow you to integrate both logical and physical backups into your backup system.

1. Oracle8 also comes bundled with a stripped-down version of Legato NetWorker, which can be used in conjuction with rman to back up to disk. However, this does not replicate the functionality of Informix's ontape or Sybase's Backup Server, both of which can back up directly to a tape or disk file without the intervention of a third-party product.

Get Every Instance

Earlier in this book, we talked about how your backup programs should be written in such a way that everything in your system is automatically discovered and backed up. Adding a filesystem should not require you to edit your backup scripts. This goes double for databases, which tend to be added and deleted much more frequently than filesystems.

You need some way to ensure that every database instance on every server is being backed up. The free utility hostdump.sh, discussed earlier, backs up all the filesystems on the box by looking at the fstab file, which lists all filesystems. Wouldn't it be nice if you had such a file for databases? Oracle and Sybase already do. Sybase has the interfaces file, which lists every server on each system. If an instance is not listed in this file, users cannot connect to it. Oracle offers the oratab file, which accomplishes the same task, but its use is not mandatory, as Sybase's interfaces file is. Some sites don't use the oratab file because they have only one instance. The best way to enforce the use of the oratab file is to write startup scripts which start up only the databases in oratab.

Informix has no file that stores all of the Informix instances on a server. This is disappointing, since many companies run more than one instance of Informix. The good news is that you can make your own inftab file, which looks a lot like the oratab file and accomplishes the same thing. Again, the way to enforce its use is to write startup programs which start up only the instances in inftab.

Since the files described here are not always used and yet should be, we would like to emphasize what we just said: you really need a centralized file that lists all the instances on the server. You should use that file to determine what instances on a given server need to be backed up. Sybase already has the interfaces file and enforces its use. Oracle has the oratab file, but its use is optional. You can create an inftab file for Informix, but its use also would be optional. Enforce the use of the oratab and inftab files by writing startup scripts which start up only the instances listed in those files.[2] A wayward (or busy) database administrator could create an instance and even get it running without putting it in this file. But if you reboot the box enough times, she will be sure to put it in the startup file eventually, so that she doesn't have to start it manually every time!

Transaction Log Dumps Are Not Incremental Backups

This important topic is often misunderstood. The confusion stems from the Sybase documentation, which often refers to a transaction log dump as an incremental backup. They are not the same thing!

What is the difference between the two? An *incremental* backup is a special backup that contains only the changed pages (blocks) since the last higher-level backup. A *transaction log dump* is a backup of all the transactions that have occurred since the last transaction log dump. They may sound similar, but they're not. The latter is much more difficult to manage and much slower to read.

2. Often all that's needed is a slight modification of the default startup scripts that come with the database.

Perhaps the best way to illustrate this would be to discuss Informix's ontape program, which can do both incremental and transaction log backups. Suppose you created a level-0 (full) backup on Friday. During the week, you did not perform any full backups but ran only continuous (transaction log) backups. Now suppose that it is Thursday and you need to restore your database. You would have your full backup from Friday and your continuous (transaction log) backups from each day. To restore, you would need to read your full backup and then read each of the continuous backup volumes in order. Assuming you made one backup volume per day, you would need seven volumes.

Now assume the same scenario, except that you also ran an incremental level-1 backup every night. If you needed to restore the database on Thursday, you would need the full backup from Friday, the latest incremental backup volume (i.e., Wednesday's), and the continuous backup volume for Thursday—three volumes instead of seven, because the latest level-1 incremental backup contains all changes since the level-0 backup.

Besides the difference in complexity, reading an incremental backup is also much quicker than reading a transaction log backup—ask anyone who has rolled through several days' worth of transaction logs. In one benchmark that one of us performed, reading two weeks of transaction logs took 36 hours. Reading an incremental backup covering the same time period took only one hour. Why? A given page may be changed several times within a short period, so replaying the transaction log changes it several times. Loading a true incremental backup changes it only once, to its most recent value.

Sybase's Backup Server, Microsoft's SQL Server & Exchange, and Oracle7's EBU have no concept of this type of incremental backup. Informix's ontape and onbar do, and Oracle8's rman does have some incremental backup capability.

Do-It-Yourself: Creating Your Own Backup Utility

You don't have to use a high-priced commercial utility to back up your databases. They certainly can make your backups more automated or centrally controlled, but since most of them run $3,000–$8,000 per system, many people are using homegrown systems.

Intermediary Disk

This is one of the most popular ways to do homegrown database backups. It's fast, clean, and easy. The basic idea is to use a script which backs up the database to disk. That backup is then treated as a regular file by the nightly filesystem backup. You can reduce the amount of disk space needed by compressing the backed-up file. If you're really pressed for space, and you're running UNIX, you can use named pipes to compress the backup as it's being written. In that case, you need a backup disk that is only one-third to one-half the size of your original database disk (depending on the compression rate you get). Unless you have a very large database, this will probably be cheaper than buying a commercial utility to perform this task. Each of the vendor-spe-

cific database backup chapters in *UNIX Backup & Recovery* contains a script you can use to do this.

Dedicated Backup Drive

Somewhat more complicated homegrown backup scripts can back up to a dedicated drive. Depending on the size of your database, this may be more or less expensive than backing up to disk, but it will definitely be slower. It is also more complex, since you must keep track of each volume and label it in such a way that you know which database was backed up to it. (If you back up to disk, this can be done by giving the backup file the same name as the database.)

Shell Scripts

We assume that you are doing the preceding backups with some sort of shell script. Shell scripts are much better than having a simple cron or at entry which says, "Back up database A to device B." Shell scripts can do lots of error checking and can be told to perform such actions as notifying the database administrator if they encounter a problem.

Calling a Professional

This is one of the biggest growth markets in the backup product industry. Most commercial filesystem backup products now have interfaces to back up your database automatically to volumes that are managed by their product. It's really beautiful, but it does come at a price! Some of these products also have an interface to a third-party program (e.g., SQL Backtrack) that accomplishes the same task.

The Big Three

Each of the three biggest UNIX database vendors, Informix, Oracle, and Sybase, has a backup utility that can interface with commercial backup products, also referred to as *storage managers*. On a high level, these backup utilities all work in essentially the same way. The database vendor's utility generates one or more backup streams via an API to which storage managers can talk. The companies that produce the storage managers can then write a utility that interfaces between their storage manager and the database backup utility's API. Remember, though, that the database backup utilities come bundled with the database products, but the commercial backup products' utilities cost several thousand dollars each.

Of the three main database vendors, only Sybase offers a backup utility that can perform backups without interfacing with a commercial storage manager. Oracle's rman and Informix's onbar both have advanced capabilities, but without a storage manager the tools are essentially useless. Since Oracle and Informix didn't want to force their customers to buy a storage manager or the interface to their backup utility, they came up with a compromise. Both of these vendors now bundle a free, stripped-down version of Legato NetWorker and its Business Suite Module with their product. This OEM version of NetWorker has significantly less functionality than the full-featured

version, but it allows you to use rman and onbar to do backups. Oracle uses rman to interface between the database and the storage manager. NetWorker communicates with the backup media and its own Business Suite Module, and the Business Suite module interfaces between NetWorker and rman. The backup data flows from the Oracle database, through rman, through the Business Suite Module, through NetWorker, to the backup media. Restores, obviously, flow in the opposite direction.

Each of the three big databases's utilities has its own backup and recovery history.

Informix

Informix has always been the easiest database to back up and recover, using ontape (formerly called tbtape), a standalone backup command designed to back up to tape. ontape is simple, has incremental capabilities, can back up to disk as well as to tape, and backs up the database live. Some of these features, which were always assumed by Informix users to be present in other database systems, are only now appearing in other products. Informix now also offers onbar, which is designed specifically to send a stream of backup data to a commercial product. Some backup vendors have ported to the earlier ontape command, while others waited for onbar. Whichever command you use, you can recover individual dbspaces.

Oracle

Historically, Oracle did not have a true backup utility, but it did have commands that allowed you to write your own—even allowing you to do live backups. Now Oracle7 comes bundled with the Enterprise Backup Utility, EBU, and Oracle8 has Recovery Manager, rman. Both are designed to send streams of backup data to a commercial backup utility. Both require a storage manager, but Oracle8 now comes bundled with a stripped-down storage manager. Some storage manager vendors have stayed away from EBU and rman interfaces, citing reasons such as performance or flexibility. BMC's SQL Backtrack is probably the best known of the commercial products. SQL Backtrack can do a true incremental backup, as discussed earlier. (The ability to do incremental backups is now provided by Oracle8's rman, but as of this writing there are still hundreds of thousands of Oracle7 databases out there. The only way to do a true incremental backup of an Oracle7 database is to use a product such as SQL Backtrack.) Whichever utility you use, you can recover individual files or tablespaces.

Sybase

Sybase has come a long way in the backup arena but it still has a long way to go. The dump command used to be very slow, severely impacting database performance, but that problem was fixed in System 10. The fix was done by separating the dump processes from the database engine and creating the Backup Server. The Backup Server also now has a nice ability to stream data to multiple backup devices simultaneously; its main problem now is that it is an all-or-nothing utility. You cannot recover an individual dataspace or device; you must restore the entire database or nothing at all. This means that if you have a 500GB Sybase database and you lose one 4GB disk, you have

to restore the entire 500GB. (It was this lack of functionality that created the market for BMC's SQL Backtrack.)

Restoring an RDBMS

The process of restoring an RDBMS varies according to the backup method you used, of course. How you proceed to restore is based on the status of your nondata and data disks and whether you are able to do partial restores online.

Loss of Any Nondata Disk

A "data" disk is defined as any disk that contains a database object. If you keep your database objects intact, you don't need to restore the database, you just have to restore missing parts that make it work! This can range from a restore of a single database setup file to a complete restore of everything on another system.

Executables

Your database can't work if its executables aren't there. This part of the recovery is much simpler if you have all your executables located in a special filesystem.

Setup files

In a pinch, you can restore the executables by copying them from a known good system, but that won't work for your database setup files. Each instance often has an initialization file that sets up certain variables such as the instance name and the location of the master database. These setup files can usually be recreated if you have logs that tell you how you made them the first time, but it is probably easier to recover them from backup.

Customized OS files

Databases often require you to edit system configuration files such as /etc/system. Changes to these files might include customizing shared memory or changing the TCP port over which software will communicate. If you are restoring onto a brand-new install or onto another system, those changes will need to be repeated, and often changes to your OS files are forgotten or poorly documented. Unless you prepared for this situation, it's probably easier just to follow the installation instructions for a standard installation. If you're reading this in advance of such an outage, however, now is the time to document all changes to config files. If you know what files are typically changed, you can even write a program that automatically documents them for you.

License setup files

Database products have not typically used heavy licensing enforcement systems, but this will probably change over time. If yours does use such a system (e.g., FlexLM), you need to restore those files before your database will function.

Loss of a Data Disk

The complexity and difficulty of your restore can vary greatly depending on which data disk you lost and how well you prepared for such a loss.

Master database

The complete loss of a Sybase Informix *rootdbs*, or Oracle control file is very difficult to recover from—so much so that you need to ensure that it never happens. (You'll be sorry if you don't!) Mirror your Sybase master database. Mirror your Informix root dbspace. Mirror your Oracle control files. Just do it. Even if you can't afford enough disk to mirror anything else, mirror this. It doesn't take up that much extra disk space, and the time and frustration you will save yourself are immense. This is the easiest way to save huge amounts of time in a major restore.

Other databases

Unless you are using Oracle (which has a one-to-one database-to-instance relationship), you may have multiple databases within an instance. If you lose only one device within a database (other than the master database), recovering is not too bad. You can probably leave the rest of the instance and any other databases online while you are doing the restore. The best preparation you can make for restoring single databases is to document where they are and what devices they consist of.

Backups

One of the most popular methods of backing up a database is to save it to disk and then allow the filesystem backup program to put it onto a backup volume. This method is very efficient, but it does have one drawback. Suppose you lose a data disk and the disk on which you store the backups. You must first restore the disk backup file from the backup volume, then restore the database from the disk file. If this two-step procedure is required, it will take longer than recovering straight from a backup volume.

Transaction log backups

When you lose your transaction logs—from dump tran in Sybase, ontape -c in Informix, or the archived redo logs in Oracle—you will need to restore all that were made since the last full or incremental database backup. Before you start a large restore, check the time that the last database backup was made, so that you can restore only the transaction log backups after that date. If you don't, you may find yourself restoring ones you don't actually need.

> If you do database backups infrequently, you may need quite a few transaction log backups to complete the restore. While restoring them, you must be careful that you have enough space free. You may have to restore them into an alternate location or compress them. You can then move or uncompress them a few at a time as the restore program asks for them.

Online transaction logs

This is where you can suffer data loss. Even if you have a good backup and all the transaction logs since the last backup, you can be in trouble if you lose the data with the online logs in it. This would be the disk with Informix's logical log or Oracle's online redologs. (Sybase stores the online transaction log in the master database.) One way to prevent this tragedy is to mirror the log.

Online Partial Restores

Some databases allow you to bring part of the database online while leaving one dataspace or datafile offline. This partial availability may allow you more time to complete a difficult restore or reduce the overall impact to your user community. This is especially true if the portion of the database that you are restoring contains a table that is not accessed frequently. Before beginning such a restore, though, consider these factors:

Interdependency of data within the database

A table containing data that is accessed infrequently may be a perfect candidate for a partial restore. You also might consider such a restore if the rest of the database can function normally (or in a slightly diminished capacity) without the table. However, suppose the database is the sales database, and the table contains all of the actual sales transactions. The rest of the database is fine (e.g., names, phone numbers, addresses). Without the table of transaction data, though, the database is useless. Since doing a partial restore increases the overall restore time, a partial restore would be a bad idea in this case.

Physical relationship between tables and dataspaces

Most database backup products do not allow you to restore on the table level. They sometimes allow you to restore on the dataspace level, however. Suppose you have lost one disk. You need to recover the dataspace this disk resides in, right? Suppose you have a partitioned table which resides in multiple dataspaces. That would mean that the entire table would be unavailable. If that table is not needed for normal operation, as described before, you might consider a partial restore. Again, however, remember that a partial restore increases your overall restore time.

Time requirement for a complete restore

Some environments don't want to hear about partially functioning databases. "Tell us when it's up. Don't say it's almost up or partially up, just tell us when you're done!" These environments are more concerned with overall downtime and should be treated accordingly. You need to restore the database in the fastest way possible. That probably would be to shut down the database and restore the affected dataspace.

Documentation and Testing

Restoring an RDBMS will probably be the most complex procedure you will ever undertake, and usually you have to do it under intense time pressure. The users want the database up now and don't really care that you've never done this before. You need to document and test your database backup procedures often to make sure that they work. The following guidelines may help:

- Set up a standalone machine with no other databases on it. It doesn't have to be large or even dedicated to this purpose, but it would be good if you could reinstall the OS as a test. When you buy a new machine, before you put it to work for real, try doing some test database restores on it.
- When you test your restores, test the worst-case scenario. Make sure that you know how to install the product onto a new system and that you know all of the files you need to edit to make it work. This is why you should be doing this on a virgin operating system: you will be forced to edit /etc/system, for example, instead of forgetting it without penalty because it's already been edited.
- You could set up a test instance and then back up and restore that. However, it would be better to take a normal database backup from one of your production systems and try to restore it to the test system.
- Provide detailed enough documentation that anyone with good sysadmin or database admin skills could follow it. If possible, do not have the person who wrote the procedure perform the test. This is the perfect task for a consultant: see whether people who know what they are doing but don't know your environment can follow the procedure.
- Have all your database administrators participate, whether they've got time or not! The worst thing that can happen is that one or two people know how to restore the databases. Then when a database crashes, one of them is sick and the other one just quit or is on vacation in the Bahamas. (One of us once saw a restore go on for hours because the database administrator didn't know he had to press Return! He called, saying, "Man, this thing is taking forever!") Make sure every database administrator knows how to restore your databases!

4. Backup Hardware

We've now looked at the reasons for backup, what needs to be backed up, and some of the software tools to perform the tasks. Now we must determine where to place all this backup information. Storage is the heart of the matter here, and our selections are numerous. We can back up to ever less expensive disk devices, optical platters, or magnetic tape of various flavors, and we can use robotic automation to handle the media for us. You can find a table comparing many of these technologies at *http://www.storagemountain.com.*

Disks

Single Disks

Disk devices are now running a penny or two per megabyte. Their dramatic decrease in cost over the past fifteen years has been accompanied by huge gains in the capacity and reduction in the access times for data. Recording technology has allowed higher densities of bits to be packed onto more exotic substrates for fast access. Disks are the second fastest medium for data storage. RAM is faster, but also costs more, at 15 to 16 cents per megabyte. A recent addition to the picture, solid state disks using non-volatile RAM, are extremely fast, but also extremely expensive. Disks, then, allow inexpensive storage of large amounts of data with relatively easy access.

RAID

A Redundant Array of Inexpensive Disks (RAID array) can combine disks in defined paradigms to provide both capacity and security for recorded data. A few of these paradigms are:

- RAID level 0, data striping: Data is written across a set of disks in parallel. For example, a 128KB chunk of data may be written in parallel on 2 disks by writing 64KB on each simultaneously, improving both performance and capacity.
- RAID level 1, data mirroring: Data is copied in parallel across a pair of disks. In the above example, 128KB is written twice, on two different disks. This protection against disk failures comes at the cost of doubling the number of disks used.
- RAID level 0+1, striping plus mirroring: Data is striped over *n* disks and

which are then mirrored on *n* separate disks. Here we gain performance and security at the cost of *2n* disks to hold data. (But, remember, disks are inexpensive, right?)

RAID level 5, striping plus parity: More security is provided by using pieces of disk to create parity data, using 1 bit per byte of extra information. So every 8 bytes of data is accompanied by one extra byte of parity. The data is striped as in RAID 0, and the parity data is interleaved. This very secure method will continue to operate in the case of a disk failure in the parity group (albeit with somewhat degraded performance).

Advanced RAID Storage

Very large arrays of disks can be joined together by products such as in EMC's Symmetrix or Hitachi Data Systems' Lightning line. These arrays are available in the dozens of terabytes, but they cost anywhere from several hundred thousand to a few million dollars, depending on the capacity and features included in the configuration.

Fibre Channel ATA RAID

A recent addition to the market is the RAID arrays built using off-the-shelf ATA/IDE disk drives connected to a SCSI or Fibre Channel controller. These extremely inexpensive RAID arrays can be used for low-intensity applications such as backup. Many people now suggest placing one of these inexpensive RAID arrays in front of your tape library and backing up to it first. Offsite tapes can then be created from this disk backup, while onsite recoveries come straight from disk.

NAS filers such as Network Appliance's NearStore product allow you to use NFS or CIFS to mount these inexpensive arrays easily to multiple systems. If you create one or more directories for each system to write to, you in effect give each system its own dedicated backup device.

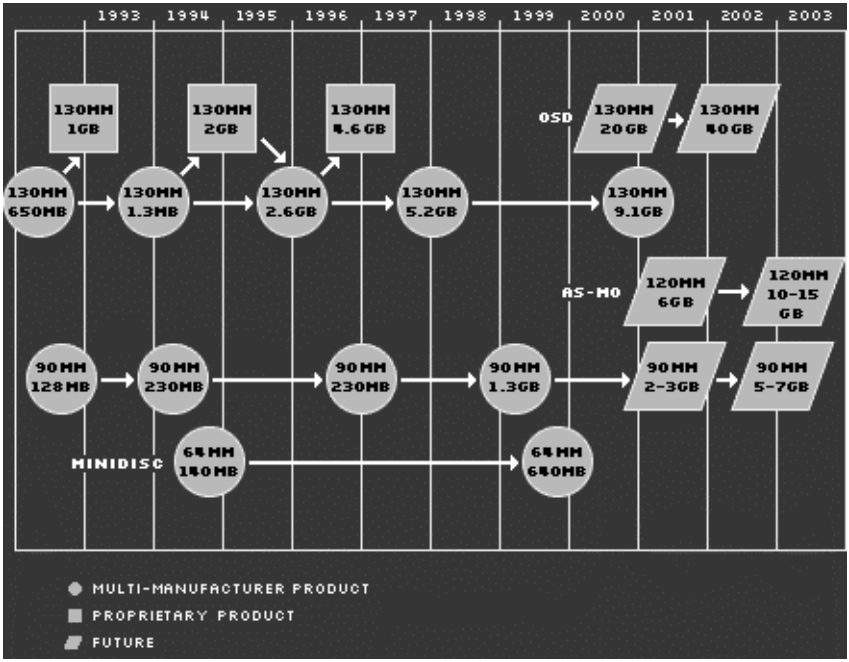
These disk arrays can be used as virtual tape arrays or libraries. They are still composed of many inexpensive ATA drives addressable via Fibre Channel, but they appear to the backup server as many tape drives, or possibly a tape library, as with Quantum's virtual P1000. They offer an 8TB virtual tape library that fits in 2U. It's comprised of 30 ATA disk drives, but appears to the backup server as a two- or six-drive P1000 with 30 slots. Requests to load, rewind, eject, or unload tapes are automatically responded to as if they were being executed, but obviously they are unnecessary. In addition, these arrays can use RAID 3, which is specifically designed for large, sequential I/O requests. This makes them an extremely fast, inexpensive alternative to tape. One downside to these disk arrays that appear as tape libraries, though, is that if you are using a commercial backup product, you have to purchase a tape library license to use them.

Optical

The middle road between magnetic storage on disk and tape uses optical methods to write data to media. A laser light source scanning a reflective surface usually serves as

the basic read mechanism. However, magneto-optical devices use a combination of laser to write and magnetic head to read the data. The capacities and formats of the drives and media vary considerably from one manufacturer to another, but there has been steady progress toward standardization. The OSTA Web page in the references provides some excellent pointers to various sites. Figure 1, from OSTA, shows the capacities and form factors for magneto-optical disk media.

Figure 1. The Development of Media



Here are some definitions and specifications:

- Erasable optical: Generally either magneto-optical or phase-change optical disks. On a magneto-optical disk, the laser changes the magnetic properties of the medium using Curie point heating of the substrate under the influence of a magnetic field. In the second, the laser induces a phase change in the substrate crystalline structure. Both mechanisms change the reflective properties of the substrate, which is how the data is read back.
- WORM: Write Once, Read Many is generally a 12-inch platter of glass that will record a large amount of data permanently. Although the published shelf life of this media is given as 30 years or more, the actual shelf life has never been tested. The problem is that the drives which read these disks have a much shorter life span and, as the technology has evolved, the manufacturers discontinue sales and support of the drive. This is thus a dead-end technology which could store around 16GB of data for a very long time.

- CD and CD-RW: The familiar CD-ROM disk is ubiquitous in the world today, holding music as well as data. However, it is limited to approximately 650–700MB of data and is read-only. The successor to this is the Read/Writable compact disk, which allows the user to erase and reuse the same physical area of the disk.
- DVD-ROM: Its basic technology is the same as DVD Video, but it also includes computer-friendly file formats.
- DVD-R: Its capacity is 4.7 gigabytes. Though similar in appearance to the CD, DVDs are written with a much smaller track size (0.74 microns vs. 1.6 microns for CD). As with CD-R, users can write only once to this disk.
- DVD-RAM: This DVD acts as a virtual hard disk, with random read-write access. Originally a 2.6-gigabyte drive, its capacity has increased to 4.7-gigabyte per side. It can be rewritten more than 100,000 times.
- DVD-RW: Similar to DVD-RAM except that its technology features a sequential read-write access more like a phonograph than the random-access mode of a hard disk. Its read-write capacity is 4.7 gigabytes per side. It can be rewritten up to about 1,000 times.
- DVD+RW: This most recent competing technology stands to make a big difference. These disks offer a higher level of compatibility than their predecessors. As of this writing,no write-once medium is available for DVD+RW drives, but that should change soon.

That pretty much covers random-access backup media. In summary, they have reasonable capacities, somewhat high prices, and variable shelf lives. These drawbacks are mitigated by the speed with which data can be accessed and read back into the system. Never underestimate the power of random access.

Tape

With tape drives the price per megabyte starts to plummet, but the access time to the data is correspondingly greater. Tape formats can be divided into three major categories—linear, helical scan, and serpentine—but they all have a plastic substrate coated with a thin film of magnetic material. The electronics generally write the data, read what was just written, and correct if necessary. Sometimes the tape moves over a stationary head and sometimes the head is a rotating device (containing multiple read/write elements.

Linear

Perhaps the simplest design and pre-dating the other tape formats, linear recordings are represented by the venerable 9-track tape drives. Whenever an old sci-fi movie wants to show a busy computer room, filmmakers invariably focus on a spinning 9-track tape. This tape method uses a fixed head over which the tape is dragged.

Writing to 9-track tape is very simple: 9 bits are written to tape in parallel, the tape

advances, and the next 9 bits are written. Densities range from 800 bytes/inch up to 6250BPI. The full capacity of the tape is relatively small though—around 140MB at the highest density. They also tend to suffer from shelf storage without data refresh, and the data can become considerably corrupted over time.

Helical Scan

To pack more bits per inch and increase the recording speed, helical scan tapes use a rotating read/write head assembly, typically made up of 2 write heads and 2 verify (read) heads. The assembly rotates at an angle to the movement of the tape. This causes the data to be recorded in angled stripes across the tape.

Several tape types comprise this group: DDS/DAT, 8mm, AIT, and DST. DDS came from DAT (digital audio tape) drives, which were developed for the audio recording industry. As time has passed, the densities have increased, the tapes are longer, and the recording methods have changed. The current DDS standard is DDS-4, a 150 meter tape recording 20GB at 2.4MB/sec uncompressed. Most drives now build data compression built into the drive electronics. Caveat emptor: many marketeers will quote compressed capacities and transfer speeds, which may have no relevance to the data you are recording.

Next on the list is traditional 8mm tape. This technology grew out of the 8mm video camera recording industry. These mass-produced video recorders were easily adapted to data recording. Generally, the capacities range from 3.5 to 7GB with transfer speeds of 0.5 to 2MB/sec uncompressed.

Because video recording requirements are less stringent than those for data recording, Exabyte designed a new recording system for the 8mm format called Mammoth. In the original DAT and 8mm tape drives, the tape is required to take a very complex path over capstans and rollers. This stress on the tape reduces life span and accuracy. The older drives also tended to get dusty as the tapes shed minute amounts of particulate matter. Mammoth engineered around these limitations for a more robust but gentler drive that allows a thinner tape, which increased the recording capacity.

Another helical scan–based drive is the AIT, whose memory-in-cassette (MIC) feature allows meta-information to be stored on the cartridge separately from the tape. AIT, like Mammoth, is an 8mm-type format, but is a completely different type of drive manufactured by Sony and HP, Exabyte's competitors. Mammoth and AIT are incompatible, but their capacities are similar. Anecdotal evidence suggests that AIT drives are more reliable than Mammoth. The AIT-3 standard calls for 100GB native storage at 12MB/sec.

DST (Data Storage Technology) from Ampex, was derived from the data recording industry for flight testing. These 19mm cartridge tapes can hold prodigious amounts of data. The DST 314 drive can hold up to 660GB of data on a single piece of media and can transfer data at 20MB/second. Backing up large databases to a single tape seems like a reasonable usage for this technology—a good instance of considering the type of backup you need and matching the media to the need.

Serpentine

So far, we have discussed tapes where the bits line up vertically across the width of the tape. This method made sense when recording density was very low. We have also discussed tapes where the bits line up in parallel stripes at an angle to the tape's edge. This method is very efficient in writing, but not for reading, but since files recorded at the end of a tape stream will be written at the end of the tape.

A third recording technology writes small tracks of data from beginning to end and then turns around and records in a parallel track from end to beginning, repeating until the tape is full. This back-and-forth method of recording is generically called serpentine. DLT and Super DLT (Digital Linear Tape) use serpentine recording, as do the older QIC cartridge tapes. As recording heads decreased in size and the recording material improved, tape capacity increased.

QIC (Quarter Inch Cartridge) was introduced by 3M in 1972. Mass production and broad usage brought the price low. The cartridge is similar to an audio tape cartridge, in that it contains both the tape and the take-up spools. The current standard is QIC-3095, which holds 72 parallel tracks and up to 4GB of data (depending on the physical length of the tape). The tape has another standard, Travan, which currently hold 10GB native on 108 tracks (TR5).

DLT is probably the most widely used medium-range tape technology today. With a readily available supply of media, drives and libraries have made this the de facto medium of choice for many backup scenarios. DLT flavors include the DLT 2000, 4000, 7000, and 8000 drives, and Super DLT drives. These employ different recording standards. The 7000 and 8000 are widely distributed and can hold 35–40GB transferred at 5–6MB/sec, native.

LTO, another serpentine format tape, is made by a consortium of manufacturers, not including Quantum. The Ultrium drive holds 100GB, transfers data at 15MB/sec, and stores the data on 384 parallel tracks. Time will tell about the reliability, availability, and serviceability of these newer drives, but they appear to be taking a straight path to higher capacity and faster transfer rates. As of this writing, LTO appears to be giving DLT a run for its money.

Other media types are covered at *http://www.storagemountain.com*.

Robotics

Now that we've covered the drives, what about handling the media? While the tapes have increased in capacity and speed, the explosion of storage space in the world market has far outstripped that pace. The only way to back up this large dataspace is to use multiple pieces of media. Many of us older sysadmins started our careers as media relocation engineers, loading and unloading tapes from drives in preparation for user requests or system tasks such as backups. Now there are robotic devices that can handle thousands of pieces of media and contain hundreds of drives. Ah, the joys of automation!

The following list, derived from www.storagemountain.com, shows the variety of devices currently available:

- ADIC makes stackers and libraries of all shapes and sizes for all budgets. After establishing a solid position in this market, they decided to expand. They now make the largest tape libraries in the world.
- ATL makes some of the best-known DLT stackers libraries on the market. Many VARs relabel and resell ATL's libraries.
- Ecrix makes small autoloaders for use with their drives.
- Exabyte once cornered the market for 8mm stackers and libraries, and they still make stackers and libraries of all shapes and sizes.
- Hewlett-Packard is the leader in the optical jukebox field, providing magneto-optical jukeboxes of up to 1.3TB capacity.
- IBM makes a line of expandable libraries for their 3490E and 3590 tape drives that can hold up to 6,240 cartridges, for a total storage capacity of 187TB.
- Overland Data offers small DLT libraries with a unique feature: scalability. They sell an enclosure that can fit several of the small libraries, allowing them to exchange volumes. This allows those on a budget to start small while accommodating growth.
- Qualstar's product line offers some interesting features not typically found in 8mm libraries. Their design reduced the number of moving parts and added redundant, hot-swappable power supplies. Another interesting feature is an infrared beam that detects when a hand is inserted into the library. They also now make DLT libraries.
- Seagate and Seagate each has a line of small DDS stackers.
- Spectralogic now concentrates on AIT libraries. Their libraries have a very easy-to-use LCD touch screen, and almost all parts are Field Replaceable Units (FRUs). The power supplies, tape drives, motherboards, and slot system can all be replaced with a simple turn of a thumbscrew.
- Storagetek offers a line of very large, expandable tape libraries. Most of their libraries can accept any of the Storagetek DLT or LTO drives. The libraries have a capacity of up to 6000 tapes and 300 TB per library storage module. Almost all of their libraries can be interconnected to provide unlimited storage capacity.
- Tandberg manufactures AutoLoaders for their SLR line of tape drives.

5. SAN and NAS

What Is a SAN?

Before beginning an explanation of SANs, we should glance at iSCSI. As of this writing, iSCSI is gaining ground and market share, but is still very new. Although we will mention iSCSI from time to time, we will only go into detail about Fibre Channel-based SANs.

Our definition of a SAN:

> A SAN is two or more devices communicating via a serial SCSI protocol such as Fibre Channel or iSCSI.

By this definition, what differentiates a SAN from a LAN (or from NAS) is the protocol used. If a LAN carrying storage traffic uses the iSCSI protocol, then we would consider it a SAN. But simply sending traditional, LAN-based backups across a dedicated LAN does not make that LAN a SAN. Although some people refer to such a network as a storage area network, the authors do not—and we find doing so very confusing. We usually refer to such a LAN as a "storage LAN" or a "backup network." A storage LAN is a very useful tool which removes storage traffic from the production LAN. But a SAN is a network that uses a serial SCSI protocol (e.g., Fibre Channel or iSCSI) to transfer data.

A SAN is *not* network-attached storage (NAS). Whereas SANs use the SCSI protocol, NAS uses the NFS and SMB/CIFS protocols. The direct access filesystem, or DAFS, pledges to bring SANs and NAS closer together by supporting file sharing via an NFS-like protocol that will also support Fibre Channel as a transport. DAFS, however, is beyond the scope of this book.

> Note: It is very common for a NAS filer to be comprised of a filer head with SAN-attached storage behind it

SANs offer a number of advantages over traditional, parallel SCSI:

- Fibre Channel and iSCSI can be trunked, so several connections are seen as one, allowing them to communicate much faster than parallel SCSI. Even a single Fibre Channel connection now runs at 2Gb/sec in each direction, for a total aggregate of 4Gb/sec.
- You can put up to 16 million devices on a single Fibre Channel SAN (at least in theory).

- You can easily access any device connected to a SAN from any computer also connected to the SAN.

What Is NAS?

The NAS (Network Attached Storage) industry is based on selling boxes to do something that any UNIX or Windows system can do out of the box: share files via NFS, CIFS, or DAFS. That is, NAS filers share files via NFS (the file sharing protocol for UNIX) or CIFS/SMB (the file sharing protocol for Windows). How is it, then, that NAS vendors have been so successful? Why are many people predicting a predominance of NAS in the future? The answer is that they have done a pretty good job of removing the issues that people have with NFS and CIFS.

The NAS vendors tried to make NFS and CIFS servers easier to manage. They created packaged boxes with hot-swappable RAID arrays that significantly increase their availability, and they decreased the amount of time needed for corrective maintenance. Another novel implementation was a single server to provide both NFS and CIFS services. You can even mount the *same directory* under both NFS and CIFS. Some NAS vendors also designed user interfaces that made sharing NFS and CIFS filesystems easier. In various ways, then, NAS boxes are easier to manage than their predecessors.

NAS vendors have also successfully dealt with the performance problems of both NFS and CIFS. In fact, some of them have actually made NFS faster than local disk!

The first NAS vendor, Auspex, suspected that the problem stemmed from the typical NFS implementation forcing every NFS request to go through the host CPU.[3] Their solution was to create a custom box with a separate processor for each function. The host processor (HP) would be used only to get the system booted up, and then NFS requests would be the responsibility of the network processor (NP) and the storage processor (SP). The result was an NFS server that was much faster than any of its predecessors.

Network Appliance, the next major vendor on the scene, believed that the problem lay with the UNIX kernel and filesystem. Their solution was to shrink the kernel to fit on a 3.5" floppy disk, completely rewriting the NFS subsystem to be more efficient, including optimizing the filesystem for use with NVRAM. They were the first NAS vendor to publish benchmarks that showed their servers were faster than some locally attached disk.

Please note that NAS filers use the NFS or CIFS protocol to transfer *files*. In contrast, SANs use the SCSI-3 protocol to share *devices*.

SAN vs. NAS: A Summary

Table 6 compares NAS and SAN. It should clear up any remaining confusion regarding their similarities and differences.

3. All early NAS vendors started with NFS and added CIFS services later.

Table 6: SAN vs. NAS

	SAN	NAS
Protocol	Serial SCSI-3	NFS/CIFS
Shares	Raw disk and tape drives	Filesystems
Examples of shared items	/dev/rmt/0cbn /dev/dsk/c0t0d0s2 \\.\Tape0	\\filer\C\directory\filename.doc /nfsmount/directory/filename.txt
Allows	Allows different servers to access the same raw disk or tape drive (not typically seen by the end user)	Allows different users to access the same filesystem or file
Replaces	Replaces locally attached disk and tape drives. SANs also offer something new: hundreds of systems can share the same disk or tape drive	Replaces UNIX NFS servers and NT CIFS servers that offer network shared filesystems

SAN Backup and Recovery

LAN-Free Backups

LAN-free backups occur when several servers share a single tape library. Each of the servers connected to the SAN can back up to tape drives that it sees as attached locally. The data is transferred via the SAN using the SCSI-3 protocol, and thus does not use the LAN.[4] Software acts as a traffic cop. LAN-free backups are represented in Figure 2 by arrow number 1, which shows a data path starting at the backup client, traveling through the SAN switch and router, and arriving at the shared tape library.

Client-Free Backups

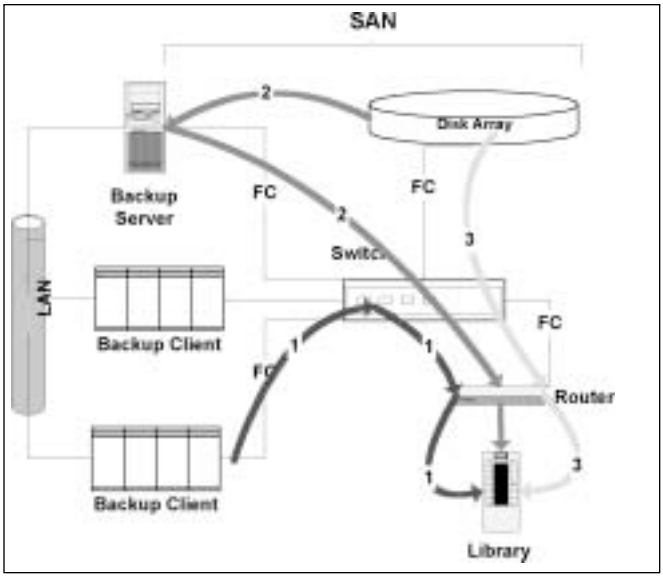
Although an individual computer is often called a server, to the backup system it is a client. In a client-free backup, the client's disk storage resides on the SAN, and a mirror of that storage is made visible to the backup server; the client is not involved in the backup.[5] Client-free backups are represented in Figure 2 by arrow number 2, which shows a data path starting at the disk array, traveling through the backup server and the SAN switch and router, and arriving at the shared tape library. The backup path is

4. This may change in the near future, since iSCSI-based SANs will use the LAN. But many experts are recommending that you create a separate LAN for iSCSI, so the backups will not use your production LAN. The principle remains the same; only the implementation changes.

5. As far as we know, W. Curtis Preston was the first to use the term client-free backups. Some backup software vendors refer to this as server-free backups, and others simply refer to it by their product name for it. He felt that a generic term was needed. We believe that this helps distinguish this type of backups from server-free backups, which are defined next.

very similar to LAN-free backup, except that the backup server is not backing up its own data, but is backing up data from another client whose disk drives reside on the SAN.

Figure 2. LAN-Free, Client-Free, and Server-Free Backups



Server-Free Backups

If the SAN to which the disk storage is connected supports a SCSI feature called *extended copy*, then the data can be sent directly from disk to tape. There are also other, more proprietary methods for doing this that don't use the extended copy command. This is the newest area of backup and recovery functionality being added to SANs. Server-free backups are represented in Figure 2 by arrow number 3, which shows a data path starting at the disk array, traveling through the SAN switch and router, and arriving at the shared tape library. You will notice that the data path does not include a server of any kind.

> No backup is completely LAN-free, client-free, or server-free. The backup server is always communicating with the backup client in some way, even if it is just to get the metadata about the backup. What these terms are meant to illustrate is that the bulk of the data is transferred without using the LAN (LAN-free), the backup client (client-free), or the backup server (server-free).

6. Summary

A good backup and recovery system is the essential starting point for a disaster recovery plan. Creating such a system is no easy task. One must understand and use native backup utilities, freely available utilities, commercial software, and plenty of hardware to accomplish this task. Recently the addition of SANs and NAS has brought new functionality to the backup and recovery space, but not without complicating the issue as well. However, a proper use of SANs and NAS in your backup and recovery system can significantly increase its usefulness and its integrity.

References

General

http://www.storagemountain.com

A comprehensive directory of storage software and hardware

Optical Storage

http://www.osta.org

The Optical Storage Technology Association

Tape Specifications

http://www.pctechguide.com/15tape.htm

A comparison of technologies

http://www.storagemountain.com/hardware-drives.html

A tabular comparison of many of the modern drives types, with vendor links

http://etoh.wopr.net/ex.abstract.html

An explanation of Extended Towers of Hanoi by Vincent Cordrey and Jordan Schwartz

SANs

http://data.fibrechannel-europe.com/magazine/articles/a_161100_01.html

An explanation of iSCSI

http://www.iol.unh.edu/training/fc/fc_tutorial.htm

A basic tutorial on SANs

http://www.nswc.navy.mil/cosip/feb98/tech0298-1.shtml

A tutorial on Fibre Channel and the Fibre Channel logical layers

Definitions

10Mb
: 10 Megabit Ethernet.

100Mb
: 100 Megabit Ethernet.

10mm tape
: 10 millimeter tape-width cartridge media.

4mm tape
: 4 millimeter tape-width cartridge media, typically DAT (Digital Audio Tape).

8mm tape
: 8 millimeter tape-width cartridge media.

ACS (Automatic Cartridge System)
: A StorageTek term.

AMANDA (Advanced Maryland Automatic Network Disk Archiver)
: A shareware public-domain backup and recovery solution.

archive
: The process of storing data on secondary media, then removing the stored data from the source system. Also, the collection of data that has gone through this process.

b
: Stands for *bit* when used with a numeric prefix.

B
: Stands for *byte* when used with a numeric prefix.

backup
: The process of saving a copy of data, typically to removable media.

backup window
: The period of time during which it is permissible to start a backup. (Generally, if the window closes after a backup is started, it will proceed to completion.)

bare metal recovery
: Rebuilding a computer system's OS and variable data on disks from square one. Bare metal recovery should definitely be addressed and covered in a disaster recovery process document.

baseline
: A full backup that will never expire.

bit bucket
: A location to send unwanted output so that it really does not take up disk space; typically, under UNIX systems it is /dev/null.

browse time
: The amount of time that a backup file can still be browsed.

cartridge tape
: Often found on workstations, they are faster and hold more information than floppies. See *QIC tape*.

CD (Compact Disk)
: A form of media that can hold data. Some compact disks are read-only (CD-ROM), and others are read-write (CD-R).

CD-R (Compact Disk—Recordable)
: Recordable media.

CD-ROM (Compact Disk—Read-Only)
: Media that cannot be rewritten.

CD-RW (Compact Disk—Read-Write)
: Media that can be rewritten.

client
: In the client-server model, the host that makes the request for a service from a server.

clone
: A duplicated entity. In this sense, an entity can be an entire system (the disk resident OS, patches, applications, variable data), specific disk data, a tape, a CD-ROM. Also, a logical set of data that has been duplicated.

cloning
: The process of making an identical copy, especially the process of making more than one copy of files during backup.

cp
: A UNIX utility for copying files.

cpio
: A UNIX utility for copying file archives in and out.

cumulative incremental backup
: A backup of all files that have changed since the last full backup, even if they've already been backed up in earlier cumulative incremental backups.

DAT (Digital Audio Tape)
: A 4mm cartridge tape backup technology.

DDS (Digital Data Storage)
: The standard to which DAT tapes must conform.

degauss
: To demagnetize a magnetic tape or other magnetic material by use of a degausser.

destructive read
: A read operation in which data is taken from a location to another destination in such a way that the data in the original location is lost or mutilated.

differential incremental backup
: A backup strategy of all files that have changed since the last differential backup.

disaster recovery
: A procedure for a host, network, or enterprise to restore systems and data following a disaster such as a disk crash, fire, earthquake, storm, or water damage.

disk
: Magnetic disk.

disk drive
: The transport mechanism of a magnetic disk unit, which moves the magnetic medium.

diskette
: Floppy disk.

disk file controller
: A device which controls the transfer of data between magnetic disk units and main memory.

disk pack
: A removable assembly of magnetic disks easily placed on and removed from a disk drive unit.

DLT (Digital Linear Tape)
: A common tape format used for high-speed data transfer, holding 35–100GB of uncompressed data.

drive
: Any device that moves a medium used for reading or writing, such as a disk drive, tape drive, CD drive, or DVD drive.

driver
: Software that controls an internal or peripheral device, such as a disk drive, tape drive, CD-ROM drive, robot, or stacker.

dump
: A BSD-based UNIX utility that writes the contents of a file, files, directories, filesystems, or disk drive.

dump cycle
: The schedule according to which data is backed up. This cycle usually involves a sequence of *full* and *incremental* dumps. Also, a term used in AMANDA to set the frequency of full dumps.

duplex
: Bidirectional communication. Full duplex uses media both to transmit and to receive simultaneously. Half duplex uses separate media or alternates direction.

duplication
: Having more than one copy (as in more than one copy of data). Also, the process of copying data (typically from tape to tape). See *clone*.

enterprise backup solution
: A documented plan for regular backups of all an enterprise's important data. It also addresses network, security, and scalability issues.

EOF
: End of file.

EOR
: End of run.

EOT
: End of tape.

EPROM (Erasable Programmable Read-Only Memory)
: A read-only memory that may be both programmed and erased in the field. Erasure is often carried out in strong ultraviolet light. Also, Electrically Programmable Read-Only Memory (electrically erasable).

erasable storage
: Any storage medium in which new data can overwrites previously written data.

floppy disk
: Magnetic media that cannot hold much data (only up to 2.8MB). They are more expensive than other media per megabyte; since they only last for a couple of years, they should never be used for long-term storage.

floptical
: Media that is written magnetically but read optically. They are the same physical size as a 3.5-inch floppy but hold anywhere from 60 to 200MB. This technology is fairly new, so it is unclear how long data can last on such media and unclear how reliable the media and associated drives may be.

full backup
: All files in the specified target source locations are saved. See *incremental backup*.

full duplex
: See *duplex*.

Gb (gigabit)
: Loosely, one billion bits; precisely, 1,073,741,824 bits.

GB (gigabyte)
: Loosely, one billion bytes; precisely, 1,073,741,824 bytes.

GBIC (Gigabit Interface Converter)
: Hardware used to attach devices to fiber-based systems such as Fibre Channel.

giga-
: Prefix meaning one billion; 10^9 or 1,000,000,000; in computer terminology, 2^{30} or 1,073,741,824.

hot backup
: A backup that takes place while the system is in use.

HSM (Hierarchical Storage Management)
: A system that stores and manages data in a hierarchical fashion. This type of system allows both backup up and restore of data and archival of data.

IDE (Integrated Drive Electronics)
: A type of hardware interface commonly used to connect peripherals to personal computers; its use is growing on some makes of UNIX hardware, such as Sun Microsystems Ultra line.

incremental backup
: A backup of data that has changed since a specified reference point. The reference point can be either a time or a certain type of backup (e.g., a full backup). See *differential incremental backup*; *cumulative incremental backup*.

Internet
: The computer network that stemmed from the original ARPAnet funded by the DoD during the late 1960s and early 1970s. The geneology of the Internet is ARPAnet, DARPAnet, NSFnet, and now the Internet. This computer network is not governed by any one body but is made up a multitude of privately controlled, globally connected networks.

intranet
: A private set of interconnected local area networks. Also, the entire set of interconnected networks for an enterprise (company); the *corporate network*. This network may or may not be connected to the Internet. See *Internet*.

IP (Internet Protocol)
: A protocol. which works at the third layer of both the TCP/IP and OSI network models and deals with getting network data sent to the correct destination.

IP Address
: An Internet protocol address, required in order for a host to be attached to a TCP/IP network. This address is represented by 4 octets (bytes) separated by dots (periods), for example, 131.106.3.253. Also, Intelligent Peripheral Interface, which was developed to be a high-speed replacement for the SMD protocol but was overtaken by the SCSI protocol.

jukebox
: A storage device that can hold several media, either magnetic (tape or disk) or optical. Such a device takes its name from the old-style record-playing jukebox, which changes records on a single turntable. Jukeboxes can automatically change removable media in a limited number of drives. They are available for several types of media, including WORM, 8mm, and DAT. Often jukeboxes are bundled with special backup software that understands how to operate the changer.

kilo-
: Prefix meaning one thousand; 10^3 or 1000; in computer terminology, 2^{10} or 1024.

LAN (Local Area Network)
: A network of computers in a relatively small area.

LAN-free backup
: A backup that writes to media on units attached to the host via a SAN.

library
: A physical device (typically, a peripheral) that holds a multitude of media such as 4mm DAT, 8mm tape, DLT, or CD-Rs. Also, the set of media, typically cataloged in some fashion and filed in order; for example, a library of old backup tapes, or a library of offsite backup tapes.

live backup
: A backup of data while users are working on the system.

local director
: A network device that performs load balancing of incoming network traffic to a set of servers.

logical volume
: Disk space that is seen by the system as being contiguous although in fact it may be comprised of non-contiguous sections of one or more physical disks.

LVM (Logical Volume Manager)
: A utility that manages the logical volumes on a computer system. The manager usually has both a GUI and a command line mode of operation.

MAC (Media Access Control) address
: Synonymous with *Ethernet address*.

Mb (megabit)
: Loosely, one million bits; precisely, 1,048,576 bits.

MB (megabyte)
: Loosely, one million bytes; precisely, 1,048,576 bytes.

media (plural; sing. *medium*)
: The entities to which data will be written or stored, e.g., magnetic tape, magnetic disks, compact disks, digital video disks.

medium
: The smallest discrete physical storage unit; e.g., a tape volume, optical platter, physical disk drive, or logical volume.

mega-
: Prefix meaning one million; 10^6 or 1,000,000; in computer terminology, 2^{20} or 1,048,576.

micro-
: Prefix meaning one millionth; 10^{-6} or 1/1,000,000.

migration
: The movement of files to deeper levels within HSM systems; implies the probable return of the file to its original location.

milli-
: Prefix meaning one thousandth; 10^{-3} or 1/1,000.

mirror
: A duplicate (mirror image) of data at a separate disk space; as data is written to one location (the primary mirror), the same data is written to a second location (the secondary mirror).

mirroring
: Writing the same data on two or more disk spaces.

multiplexing
: The process of using several plexes to make one logical volume. See *plex*. Also, the process of combining multiple data streams into one stream; used to interleave backups from multiple filesystems onto one tape.

nano-
: Prefix meaning one billionth; 10^{-9} or 1/1,000,000,000.

NAS (Network Attached Storage)
: A computer or device dedicated to sharing files via NFS, CIFS , or DAFS.

networker
: The command to invoke the Legato Networker backup and recovery application GUI.

newfs
: A UNIX utility that puts a UNIX filesystem onto disk media.

NIC (Network Interface Card)
: A hardware card put into a computer to enable network connectivity. This board may or may not contain its own Ethernet address.

node
: A component in a network (e.g., client or server). Also, an element in a tree.

offsite storage
: Storing copies and/or older versions of backup media at an external (offsite) location, so that computer systems may be rebuilt after a disaster. This offsite location can be a different building within the work complex, a third-party vendor who specializes in storing data media, or a company building in a different city.

optical media
: Media that use optical technology (as opposed to magnetic technology), e.g., CD-ROM, CD-R, DVD-ROM, DVD-RAM, DVD-R, DVD-RW, DVD+RW.

Oracle
: A vendor of shrink-wrapped database software.

Oracle database
: The database generated and maintained by the software from Oracle. Oracle databases typically require orderly shutdown prior to a backup or recovery of the database.

oratab
: The configuration file for an Oracle installation.

parallelism
: The ability to handle more than one task (in this booklet, backup and recovery) simultaneously. See *multiplexing*.

peripheral device; peripheral
: A device attached externally (i.e., peripherally) to a system.

peta-
: Prefix meaning one quadrillion; 10^{15} or 1,000,000,000,000,000; in computer terminology, 2^{50} or 1,102,150,455,682,648.

pico-
: Prefix meaning one trillionth; 10^{-12} or 1/1,000,000,000,000.

pipe
: A standard feature of UNIX shells and DOS which allows the output of one command to become the input for another command. Usually represented as the vertical bar, |.

plex
: A logical grouping of subdisks.

QIC (Quarter Inch Cartridge) tape
: A magnetic tape technology used for backup. They can be quite expensive and are not well suited for backups, as a small filesystem will take several tapes, but are good for distributing software and for exchanging small amounts of data among different platforms. The most common cartridge tapes are QIC-11, QIC-24, and QIC-150 (the number specifies the capacity of the media in MB).

RAID (Redundant Array of Independent Disks)
: A set of disks combined in defined paradigms to provide both capacity and security for recorded data.

RAID Level 0
: Striping; not really RAID because there is not redundancy. See *striping*.

RAID level 0+1
: Striping plus mirroring, in that order. See *striping*; *mirroring*.

RAID Level 1
: Mirroring.

RAID Level 1+0
: Mirroring plus striping, in that order. See *mirroring*; *striping*.

RAID Level 2
: Striping across disks with a separate disk set used for Hamming Code Error Correction Code storage.

RAID Level 3
: Striping with one entire disk in the set used for holding parity (XOR).

RAID Level 4
: Another parity code recording scheme, similar to RAID 3.

RAID Level 5
: Striping with parity spread across all disks: if one disk dies, the system can run (with degraded performance) by getting the data from the lost disk from the parity spread across the other disks. A new disk can be hot-swapped for the dead disk, and the data from the parity will regenerate the new disk.

RAID Level 5+1
: RAID Level 5 (described above) plus mirroring.

rdump
: A UNIX utility that allows dump to be performed over the network.

recovery
: The process of restoring the data saved during a backup.

recover
: A Legato Networker client-side command line command for performing data recovery from magnetic tapes written during the backup (or save) process.

relocation
: Movement of a file or files from one filesystem or directory structure to another; implies a permanent change of file location.

resident
: Indicates that the object is present on the medium (level) being examined.

restore
: A UNIX utility that restores a dump file. See *dump*. Also, the command that invokes the UNIX utility restore./ See *ufsrestore*.

retention time
: The amount of time a backup file is considered valid.

robot
: The mechanical device within a stacker, jukebox, or library that moves media (typically tapes) in and out of the drives.

SAN (Storage Area Network)
: Two or more devices communicating via a serial SCSI protocol such as Fibre Channel or iSCSI.

SCSI (Small Computer System Interface)
: A hardware interface used to connect peripherals to a board that connects to a computer's motherboard.

SCSI over IP
: The SCSI protocol running over the third layer of the TCP/IP protocol stack.

server
: A host on a computer network that provides a service to the network or a subset of host on the network.

server-free backup
: Also known as SCSI-3 extended copy: a host asks 2 SCSI devices to transfer data directly between themselves without generating interrupts for every read/write operation on the host.

Shared Storage Option
: Software used to share tapes, libraries, and robots in VERITAS NetBackup.

Shoeshine
: What occurs when a tape drive has to stop writing, reposition, and resume writing. This is usually a symptom of data-stream starvation, i.e., the drive could handle much more input with higher write efficiency.

SMD (Storage Module Device)
: An interface that used to be popular on UNIX systems. An SMD controller can handle up to four disk drives.

snapshot
: The image of a computer system in a certain state, captured so that the computer system can be restored to that state.

stacker
: A simple tape changer, used with a standard tape drive. Tapes are loaded into a hopper; full tapes are ejected from the drive and replaced with blank tapes from the hopper. Most stackers hold about ten tapes.

stage in
: To recall from a deeper level of HSM and make resident.

stage out
: To migrate to a deeper level of HSM and remove from the source medium.

storage library
: Any robotic device that manipulates removable media volumes.

storage medium
: Any recording medium used for data backup.

streaming tape drive
: A drive capable of continuously writing data to tape at the speed the tape moves through the mechanism.

striping
: RAID terminology for the ability to write a stream of data across a set of disks in parallel.

tape
: Magnetic tape.

tape drive
: The physical unit which reads and writes magnetic tape.

tape label
: A record at the beginning and one at the end of a reel of magnetic tape which provide details about the files stored on the tape.

tape loadpoint
: The position on a piece of magnetic tape at which reading or writing can begin.

tape mark
: A character on a magnetic tape file which indicates the start of a new section; also known as a *control mark*.

tape pool
: A set of tapes grouped logically together, from which the backup application will pull during the backup process.

tar (*t*ape *ar*chive)
: The UNIX command to invoke the UNIX archiving utility tar.

TB (terabyte)
: Loosely, one trillion bytes; precisely, 1,099,511,627,776 bytes.

TCP/IP (Transmission Control Protocol/Internet Protocol)
: The standard communications protocol of the Internet.

tera-
: Prefix meaning one trillion; 10^{12} or 1,000,000,000,000; in computer terminology, 2^{40} or 1,099,511,627,776.

three-way mirror
: A mirror set made up of primary, secondary, and tertiary submirrors such that the same data is written to all three submirrors. One submirror can be split from the other two for a backup while the system is in use.

throughput
: The rate at which data can move from the system's storage media (typically, its disk drives) to the backup storage media (typically, removable tape media).

TIR (True Image Recovery)
: A feature of VERITAS NetBackup which allows recovery to a point in time but will not overwrite files created after that time.

Towers of Hanoi dump sequence
: A sophisticated data backup sequence, based on a puzzle called the Towers of Hanoi, which provides extensive redundancy. The connection between the sequence and the puzzle is limited and unimportant. The sequence is designed to balance the desire to save as much information for as long as possible against practical limitations, such as the amount of time it takes to perform the dumps and the number of tapes.

tree
: A directory structure. Also, a data structure which allows selection by reduction or expansion cascading. Its graphic representation typically resembles a tree.

ufsdump
: A System V–based UNIX utility that dumps a file, files, directories, filesystems,or disk drives.

ufsrestore
: A System V–based UNIX utility that restores a dump file. See *dump*.

vaulting
: The process of keeping a copy of the data offsite or in a secure area. See *offsite storage*.

VLAN (Virtual Local Area Network)
: Hosts on physically distinct networks or network segments that can be treated as being on the same network segment.

volume
: A generic term for a piece of media without regard to its physical form (e.g., optical disk, tape, floppy, DVD). Also, a collection of disks into a multi-disk set which can be manipulated as if it were a single large disk.

WAN (Wide Area Network)
: A network that connects a large geographic area. See *LAN*.

WORM (Write Once Read Many)
: An optical disk that cannot be rewritten.

About the Authors

W. Curtis Preston has specialized in designing storage systems for over nine years, and has designed such systems for many environments, both large and small. He is the president of The Storage Group, Inc., a storage consulting and integration company. He is also the creator of StorageMountain.com (formerly BackupCentral.com), the author of the books *Unix Backup & Recovery* and *Using SANs and NAS*, and publishes a monthly column for *Storage Magazine*.

Hal Skelly originally planned to embark on a career in graphics programming after obtaining his Computer Science degree at the University of California, San Diego. A former teacher introduced him to the world of systems administration and he has worked in that field for the past fourteen years. Hal has specialized in hierarchical storage management, mass storage, backup and recovery, and data security. He is currently supporting a government contractor administering classified systems.