The Next Step in Backup and Restore Technology

Introduction

Every advanced computer site must protect its data through some kind of backup mechanism. The data must be protected from three kinds of loss: media loss (i.e., disk failure), user error (e.g., accidental removal), and catastrophe (e.g., a tornado destroys the entire building). Note that solutions such as disk arrays mitigate only one kind of data loss. Thus, backup is a way of life for anyone using computers in their business or operation. System administrators and purchasing decision-makers must include the strength of backup as part of their system software evaluation.

Unfortunately, the classic UNIX backup programs based on Berkeley (BSD), USL (System V) or OSF system software seem to suffer from a common set of problems. These include:

- Inability to work correctly with data while in use (i.e., required offline backups are intrusive to users)
- Requirements for extensive operator intervention
- Lack of a good catalog of which files are on which tapes
- Poor end-of-media handling
- Slow backup speed
- Very slow file location on restore
- Slow restore speed
- Tape management
- Inability to sequence automatically through several local and remote data sets
- Cumbersome management of multiple tape drives, local and remote

Third party vendors (including Legato, TransArc, Delta Microsystems, General Atomics, and others) have addressed some of these problems, but none has been able to solve all of them.

This paper describes the SunSoft Online: Backup system that attacks and solves these problems.

An Analysis of the Requirements

Requirements for Backing up File Systems

The backup program is the heart of a backup system. Whether one uses "ufsdump" (known before Solaris 2.0 as "dump"), "tar," "cpio," "bar" or some other program, the goals are similar: copy a set of files (or an entire partition) to a backup medium in hopes that the files can later be read back if needed.

One major problem with backup programs, however, is that without kernel support (or a new file system), ensuring the validity of a backup tape is a difficult matter indeed. In fact, analyzing the behavior of a backup system when presented with moving subtrees inside the file system structure often exposes problems of safety. The popular "ufsdump" program, when run with a file system mounted, can get into trouble when inodes contain a directory when the first scan occurs but are re-used in the interim and become regular files. It can produce a tape which is unreadable by ufsrestore.

Of course, all online systems must contend with files being written precisely while they are being backed up. TransArc's Andrew File System product solves this problem by creating copy-on-write inodes — but this requires that an entire new file system be used on your disks. Otherwise, backup programs do they best they can, sometimes warning the operator when a file was modified during backup, sometimes silently failing.

The above remarks are not as a tirade against breaking the rules and running traditional "ufsdump" program while file systems are mounted. The probability of some of these kinds of failures is very small — nonzero, but small. The risk is often acceptable when weighed against the cost of file systems being unavailable (often for hours). Some sites have used ufsdump in online mode without noting any problems; others have occasionally bitten by taking the chance.

At any rate, it is important to make certain promises about the quality of backup tapes produced. Programs with stronger guarantees (e.g., "never creates a tape which can't be restored," "always copies every file that is on disk when the backup began") are more desirable than those with weaker guarantees ("always creates a tape").

Another problem seen too often in the world of backup programs is speed (or lack of it). Some backup programs run as slowly as a few tens of kilobytes per second — certainly too slow in a world of multiple gigabyte disk systems. While affordable 9-track tapes have ranged from 450KB/sec to 750KB/sec in speed, the capacity of 9-track tapes is rapidly falling behind other peripherals, especially in light of the frequency of operator invention required to change tapes.

Speed and Throughput Issues

Exabyte 8200 drives can achieve sustained throughput of just under 250KB/sec — slower than 9-tracks but with a 2+GB capacity. The new Exabyte 8500 drive runs twice as fast and has double the capacity — thus affording relatively intervention-free backups. Sites with 10GB of storage that desire intervention free backups can probably afford two drives. New compression devices claim compression rations that range from 2 to 5 — thus extending the capacity of a single high-density drive to between 10GB and 25GB (with concomitantly higher effective transfer rates). The backup software should, of course, be able to drive these devices at high speed.

Optical disks and quarter-inch tapes are two other backup media in use — but they're relatively slow when contrasted with 9-track drives. A backup program should be able to exploit 500KB/second drives in order to perform full backups on a 12GB disk system in eight hours or so (four hours if one has two 8mm drives).

What a New Backup Program Should Support

A new backup program should also keep catalog of files it backs up. Furthermore, the backup system should include a means to sequence through local and remote partitions (see the section on the "sequencer," below).

Additionally, a new backup program should support label verification to reduce operator error and data loss associated with mounting incorrect tapes. Of course, producing tapes that are backward compatible with the old ufsdump program would be a definite plus.

A new backup program should also reduce operator intervention by recognizing and dealing correctly with end-of-media marks. The current ufsdump program relies on knowledge about tape length in order to know when to change tapes. A program which correctly recognized end-of-media could, in fact, enable packing many file systems onto a single tape.

Any redesign of the ufsdump program should investigate implementation of other "niceties" that users have requested for many years: mail notification in case of errors, true incremental backups, and automatic device switchover. Furthermore, some sites request support for "parallel backups" in which two independent sets (fulls and incrementals) of backups are interleaved in time (i.e., on alternate days). This insulates a site against loss of data due to backup media failure. Support for this option requires a new backup program to be able to use different "dumpdates" files.

Requirements for Restoring Files

The UNIX "ufsrestore" program interface is notoriously poor. While backup tapes have an index preceding their data, tar tapes (and others) must be read completely to learn their contents.

A good backup system will support a catalog that is updated by each ackup session. Users should be able to peruse the catalog (using the familiar "cd" and "ls" commands) in order to locate versions of backed-up files and request their retrieval. Operators should intervene only to the extent required by security. Of course, the backup catalog must also protect against users seeing filenames they are not normally able to see.

While it is clear that users should be able to see information about files that is normally given by the "ls" or "ls -algs" commands, the question arises: How many versions of a file should a user see? The answer to the question should include enabling the user to:

- See the file system as it existed at any point in time
- See all versions of a given file
- See files back to the most recent level 0 backup
- See files all the way back to the first backup ever

Specifying what to restore (files, directories, an entire hierarchy) should be also simple and intuitive.

Another significant problem with ufsrestore is its speed. Many systems see a restore speed of eight hours per gigabyte of disk. This is intolerably slow — especially since situations which require large restores are precisely those in which users want to get back online quickly.

Why are ufsrestore and its brethren so slow? It's those synchronous writes. The operating system likes to make absolutely sure that file systems are not corrupted badly when a machine crashes and all the data stored in its memory are lost. The kernel carefully ensures that data from directory operations (like "create a file") is physically transferred to disk before other operations proceed. It is this synchrony that slows ufsrestore.

Providing a per-file-system switch to disable this security feature (and increase the probability of totally scrambled disks after a crash) is certainly acceptable in the world of full restores. Since data transfer rates appear to increase by a factor of about four; restarting restores from scratch after the occasional crash still results in no net loss of time.

Requirements for Simple Tape Management

Avoiding the overwriting of tapes is especially important for backups. A simple tape management system that utilizes even as simple an approach as the label field in backup headers would be a dramatic improvement. Augmenting it with a small database to keep track of expiration dates would help even more by automating the choice of "which backup tape(s) to use today."

Requirements for Backup Sequencing

Even novice system administrators regularly create customized programs and procedures for performing backups. These programs are usually simple shell scripts, awk scripts, or C code.

The scripts sequence the backups, remind the operator what's going on, and sometimes spread the workload of full backups throughout the week. Some of the more sophisticated software attempts to perform low level tape management. Other administrators use tape drives (or other secondary storage devices) hosted by other machines (e.g., IBM 370 series machines) and rely on their tape management.

The scripts attack a well-defined (and commonly known) set of problems. They attempt to:

- Minimize operator intervention. It is much easier for an operator or administrator to spend ten contiguous minutes setting up a backup than ten minutes spent one minute at a time walking to the machine room, changing a tape, labeling a tape, and re-starting the backup process.
- Sequence through file systems automatically. The ufsdump program is not particularly good at this without assistance.
- Sequence through tapes (and tape drives) automatically. Few currently available programs can save files across tapes with correct end-of-media handling.
- Automate (where possible) tape handling and labeling. Overwriting a backup tape is one of the easiest ways to lose data. Good backup systems attempt to guarantee that some minimal number of recent backups is saved and that backup sets have a minimal lifetime before they are recycled.
- Be simple and understandable. Complex schemes — particularly in multi-administrator environments — can be error-prone (in environments where errors are least tolerable).
- Support the "remote backup user" security model so that clients do not require root access on servers in order to use the servers' tape drives, and servers do not require root access on clients to backup the clients' file systems.

Some scripts go so far as to attack the problem of choosing tapes to store off-site. This is a difficult problem when, for instance, tapes contain many different backup sets.

Supplying a program to ease setup and backing up file systems is desirable for installations of all sizes. New sites (presumably with novice administrators) will benefit from dramatically eased installation and configuration. Sophisticated sites can exploit a good execution system to ensure that their backup requirements are met.

Attempting to create a single program that encompasses all the functionality of the hundreds of scripts in the world is extremely difficult. Nevertheless, it is important to be able to configure the software to emulate the most popular schemes. The richness of configuration options leads to high flexibility — and high complexity. This means that setting up the configurations becomes an error prone or difficult process in itself.

Requirements for Network Support

Of course all the above requirements should work in a networked environment with file systems and tape drives spread throughout a network. Additionally, operators should be able to interact with all the components of an integrated backup system from anywhere within the network.

The Next Step — Online: Backup

Online: Backup enhances the ufsdump and ufsrestore programs (called "hsmdump" and "hsmrestore") in addition to providing three components which provide an easy-to-configure, easy-to-run, and reliable system for backing up a site's file systems. These components include a configurator, a sequencer, and a catalog.

The sequencer interprets configuration files created by the configurator. It deduces parameters with which to invoke the hsmdump program and then iterates those invocations (potentially both locally and remotely) to save a site's files. The sequencer is extremely robust and knows how to sequence backups in the face of crashes, holidays, and unknown (frequent or occasional) invocations.

The configurator runs once to set up each configuration file and then later may run in edit mode as configurations change (e.g., adding a new disk or a new diskful system). The configurator presents a simple set of choices that administrators easily discern.

Online: Backup keeps a catalog of backed up files and information about them. The "hsmdump" program updates the catalog; the "recover" program reads the catalog and provides users with a hierarchical view of all backed up files.

The Sequencer

The sequencer runs on a machine which may have several configuration files (one for each set of backups to be performed in a single group). Each configuration file uses its own "dumpdates" file. The file systems to be backed up need not be mounted on the machine on which the sequencer runs though network communications must connect the machines. One can run as many sequencers as desired, but one copy of each configuration file concurrently. It may be advantageous for sites owning multiple backup devices (e.g., two Exabyte drives or stackers) to split their backup chores into two backup sets in order to maximize use of the drives.

The sequencer not only interprets these configuration files but also interacts with the tape database. The backup program itself knows how to verify tape labels and create new tape labels (using information supplied through command line arguments and potentially through a data file created for this purpose).

The sequencer also keeps complete logs of its activity so that administrators can gain confidence that backups are being performed as expected. The logs reside in the /var/opt/SUNWhsm/dumplog directory and include: tape request messages, all dump output, tape usage information, and error information. A typical site will see 5KB-50KB of data logged daily. Each message includes the date, time, machine, and configuration file that was running to generate the log message.

As one would expect, the sequencer performs both full and incremental backups. The sequencer, the backup program, and the file catalog support a switch that specifies true incrementals ("files changed since most recent of any level 0-9 dump") rather than the standard level 9 incremental: "any files changed since the most recent of any level 0-8 dump."

The sequencer supports cyclical backup schemes (using the level numbers of the standard dump program). These are typified by a daily sequence like "05555" which means: on Monday (which is to say "the first day"), do a full backup, then on Tuesday, Wednesday, Thursday, and Friday perform level 5 incrementals. Some file systems in a backup set may use a "50555" or "55550" cyclical backup in order to spread the full-backup load across different days ("staggering").

A more complex cyclical backup scheme includes one very similar to Epoch's baseline backup: "08888588885888858888." In this scheme, a full backup is performed only every four weeks (every 20 invocations, that is). Daily incrementals and weekly "super-incrementals" follow the full backup. Extending the sequence increases the full-backup interval — not necessarily advisable since susceptibility to tape errors and data loss is increased.

The configuration and sequencer programs run without knowledge of the precise frequency that backups are to be performed. Knowledge of holidays is complex and requires operator intervention. This configuration file design obviates the need to know anything other than "this program is being run today and should perform the next backups in sequence."

```
tapelib                   tape data
dumpmach                  index
dumpdevs                  /dev/rmt/0bn fred:/dev/rmt /0bn
block                     64
tapesup                   2
notify                    kolstad
rdevuser                  rdev
longplay
cron                      0 200 1600 0 0 1 1 1 0 1 0 1 0 1 0 0
0

#      level     multiple      days     min available
keep     0          1           15            3
keep     0         26           -1            3
keep     5          1            5            3

mastercycle 00014
fullcycle 00001 -/              >0555555555
fullcycle 00001 -/user          >5055555555
fullcycle 00001 -/mnt           >5505555555
fullcycle 00001 -/usr2          >5550555555
```

Figure 1

Sequencer Configuration File

Figure 1 depicts a sample backup configuration file. Here's how it works:

- Blank lines and comments preceded by "#" are ignored.
- The "tapelib" parameter names the tape library to use. See the section below on tape libraries.
- The "dumpmach" tells which machine owns the online catalog of backed up files. This can be the local machine or a remote machine.
- The "dumpdevs" line has a white-space separated list of tape devices to use. This list should be homogeneous in type (e.g., all 9-track tapes or all Exabyte tapes).
- The "block" line specifies the block size that backup should use.
- The "tapesup" keyword specifies the default number of tapes to "plan ahead" for potential use. This number of tapes will be "reserved" (for 23 hours) in the tape library each time the sequencer is invoked. The operator is informed which tapes are reserved and hence will be used as the backup session progresses.
- The "rdevuser" line specifies which user should be used as the login user of choice so that clients do not require root access on servers in order to use the servers' tape drives, and servers do not require root access on clients to back up the clients' file systems.
- The "longplay" specification indicates that tapes will be left in the drive between successive invocations of the sequencer (thus reducing operator intervention even more, since tapes need only be changed every several days instead of daily).
- The "cron" line contains all of the data "dumped," the dump configuration editor, needs to enable and disable automatic backups through crontab entries. (This line is not intended to be human-readable.)

The next section describes how long to keep backup tapes before they are reused. Each "keep" line has four fields besides its keyword: the level described by this line, the "multiple," the minimum number of days to keep this kind of tape tape, and the minimum number of this kind of tape to keep on the shelf before being reused. The "multiple" works like this: every backup has a "fullcycle" number (0 for the first time through the list of file systems, 1 for the second, 2 for the third, and so on). When the "fullcycle" number is a multiple of the number in the second column, then that particular column applies. A multiple of 1 represents every backup; a multiple of 2 represents every other backup, and so on.

Of course every backup tape “fullcycle number” is a multiple of 1. The sequencer figures the most restrictive set of “keep” criteria for any given tape (so each 26th tape is kept a long time, even though the “fullcycle” number is a multiple of 1 in addition to 26).

Finally, the last lines describe not only the scheme to be used for backing up the file systems but also contain state information about current and recent backups.

The “mastercycle” line tells how many times the sequencer has completely traversed this particular backup set. Its initial value is 0. Additionally, the tape database keys expiration dates off both the date and the “mastercycle” number. It calculates the longest sequence of tapes that can intervene before, say, three copies of a level 5 incremental. Adding this length to the mastercycle number gives an upper bound of a “safe” expiration cycle for tapes in the database.

Each subsequent line contains information on file systems to be dumped:

- The keyword “fullcycle”
- The number of times this particular file system has completed a full backup
- A +, -, or * with the following interpretations: + means this file system has been dumped during this cycle; - means it hasn’t; * means a backup was attempted but failed
- The name of the disk device to back up (potentially including a machine name to indicate remote backups)
- A sequence of digits (including x for true incremental) indicating backup levels
- An indicator (“>”) that shows the NEXT backup to be performed on this file system; it is advanced at the same time the “-” changes to a “+” in a given master cycle and resets to the beginning of the level sequence before pointing to the newline

The sequencer traverses the backup set using the configuration information to format backup commands and execute them. It updates the configuration file in place by moving the indicator (“>”) to show which is the next backup to perform and changing the “-” to a “+” (or “*” in the case of failure).

This sequencing scheme has several interesting properties:

- No knowledge of holidays or other service interruptions is required
- No knowledge of frequency of execution is necessary: the sequencer always does “the next right thing”
- Crash behavior is quite benign. Sequencing resumes where it left off (potentially backing up a partition twice)
- It is simple to understand and implement
- It is easily extensible to “parallel backup sets” in which two sets of dumpdates files and sequences are used to increase resiliency in the face of tape loss

Configuration

The “dumpconfig” program asks the user a few simple questions and sets up a basic initial configuration. This initial configuration can then be modified with the curses-based editor, “dumped.” The “dumpconfig” program itself takes only a couple of minutes to run. The user is then (if he or she chooses) placed in “dumped” to add remote file systems, other tape drives (local or remote), set-up automatic backup execution, or to adjust any of several other parameters as needed.

In order to make “dumped” useful for both the novice and the experienced system administrator, it has an expert mode which may be toggled on and off. When initially started, “dumped” has expert mode turned off, and only the most commonly used options are presented. For the experienced individual with special backup needs, switching to expert mode provides all of the feature-rich flexibility of the product.

Tape Management

Both the “hsmdump” program and the sequencer know how to manipulate named tapes in a rudimentary way. Falling far short of the protection and interchangeability afforded by ANSI labeled tapes, the tape library nevertheless ensures that tapes are not used before their expiration date has elapsed and that tapes are not inadvertently overwritten because of a tape selection or mounting error.

After tapes are initially used (and the labels are written), no maintenance is required.

Disaster Recovery

Once catastrophe strikes, it is important to be able to recover an entire file system hierarchy — even if the catalog of files (with the names of the tapes upon which they are kept) is lost.

To cope with this potential problem, "dumpex" copies tape library information from the database as the last backup each time "dumpex" is invoked. The tape name and file number is mailed to the notification list. Administrators are additionally encouraged to keep the "last tape written" in a special place (until a new tape replaces it). This way, the tape is easily found.

A recovery program reads the tape and instructs an administrator with the sequence of tapes to load and programs to run to recover the entire database. Once the database is recovered, file recovery is simple.

Security Issues

Security issues for any backup system revolve around access to tapes and tape drives. Access to a tape implies access to any backed up file on that tape, hence invalidating any security notion. Access to tape drives is often a weak point, since backup tapes may be left on the drive after a backup is complete. Obviously, physical access to tapes invalidates security for a site.

In the catalog of backed up files, viewing of filenames is restricted by file permissions, just like UFS file system. Users only see the catalog for the machine upon which they are executing (thus avoiding impersonation problems).

In the world of network security, Online: Backup does not use encrypted data, and thus is vulnerable to programs which use the network interface tap. Spoofing (i.e., of the operator monitor) is quite possible. No special user authentication is performed; secure RPC is not used.

Online: Backup Limitations

Highly secure customers (e.g., C-2 kernels) should not use any remote backup program. Customers with less than 1 Gbyte of disk who already have a backup scheme will benefit most from ease of frequent backups and ease of restore. Customers with difficult requirements (e.g., staging backups to empty partitions for a day, then backing up the partitions) may find Online: Backup to be inappropriate for their needs.

Conclusion

Automatic execution and sequencing of backups coupled with a rudimentary tape management system dramatically eases the operations load of performing backups. This paper describes a system that can free operators almost entirely from the daily backup burden and considerably ease the burden of file restoration.

Online: Backup

The Next Step in Backup and Restore Technology

A White Paper

SunSoft

A Sun Microsystems, Inc. Business

Portions of this paper were previously published in the proceedings of Sun User Group, United Kingdom (SUG UK), 1991.

Table of Contents

SunSoft, Inc.
2550 Garcia Avenue
Mountain View, CA 94043

For more information, call 1 800 227-9227.

Printed in USA 10/92 92266-001