LBL-35282
UC-405

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

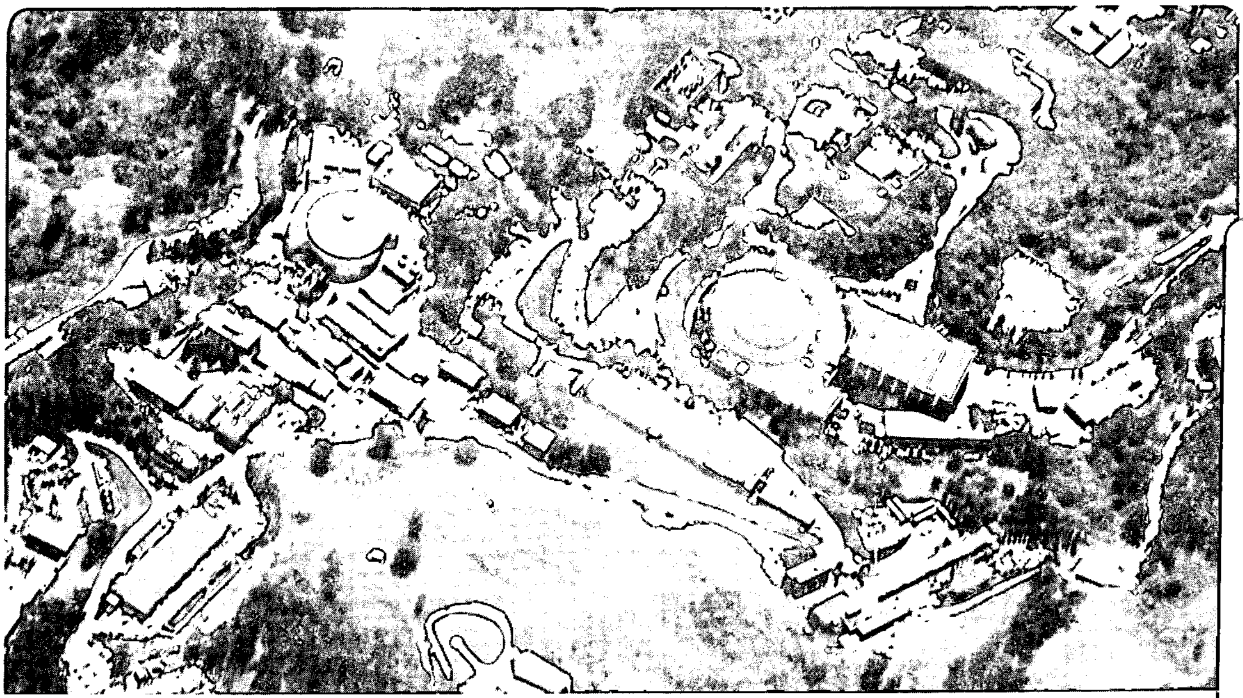
Information and Computing Sciences Division

To be presented at the Twelfth Annual Pacific Northwest Software Quality Conference, Portland, OR, October 17–19, 1994, and to be published in the Proceedings

Accountability for Network Backup Failures

W.H. Benson

February 1994



Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098

LBL-35282
UC-405

Accountability for Network Backup Failures

William H. Benson

Information & Computing Sciences Division
Lawrence Berkeley Laboratory.
University of California
Berkeley CA 94720

February 1994

This work was supported by the Director of the Scientific Computing Staff, Office of Energy Research, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

Accountability for Network Backup Failures

William H. Benson

(to be submited to Twelfth Annual Pacific Northwest Software Quality Conference, Oct 17-19, 1994, Portland, OR)

Abstract

Regular hard disk backups for workstations are widely recommended. The necessity of backups - akin to one's own mortality - is something most people would rather not think about. This attitude has two consequences. When people do subscribe to automated network backups, they expect the system to perform at a high level of reliability and that their files will be there for them when they need them. Second, they usually fail to appreciate that reliability is a shared responsibility. Although ostensibly their only responsibility is to keep the computer powered on overnight, there are actually many more opportunities for failure within the user's jurisdiction than in other parts of the infrastructure.

High reliability is almost a *sine qua non* for backups. We describe a strategy for enhancing reliability based on the principle of *accountability*. This strategy involves monitoring the system, gathering statistics, detecting problems, anticipating problems, troubleshooting, and finally determining where failure occurred within the infrastructure and who should be accountable.

We describe a specific backup system in a specific network environment to illustrate the value of accountability. This system, *macdumps*, backs up Macintosh disks over an AppleTalk network. The original software was written by Dan Tappan of BBN in the early years of the Mac and is available by ftp for non-commercial use. It has proven reliable and robust. Despite the high quality of the fundamental software, there are still many opportunities for failure within the infrasructure.

We first discuss accountability in the context of network backups, then briefly describe how the backup system operates, the components of the infrastructure, types of failures experienced, and then summarize our experience.

Accountability

In Computing and Accountability [1], Helen Nissenbaum argues that there is a tendency towards diminished accountability for harms and risks in computerized systems. She identifies four barriers to accountability, three of which can be seen at work in the context of backup failures.

collective responsibility ("many hands") - In the complex environment of network backups, it is often not readily apparent - to either the user or the administrator - precisely why a given failure occurred. The network itself - consisting of cables, routers, bridges, repeaters, etc. is a good example. The network carries unpredictable traffic loads, consists of hardware and firmware from many vendors, and is maintained by many people. It is very easy to blame a failure on an unspecified "network problem".

On the other hand, it may be possible to identify specific component breakdowns, breaks in connectivity, or scheduled or unscheduled maintenance, etc. Informing users about the specific nature of such failures lets users judge for themselves the reliability of individual components, without damaging their perception of the reliability of the system as a whole.

the mind-set that bugs are inevitable - Over the last few years of running backups, we have discovered strange and exotic bugs in very unexpected places. Two notable examples involve defective multiport repeaters - about a third of those installed on our network were affected; and a virus protection program(!) that corrupted packets on the network.

The multiport repeaters were from a certain range of serial numbers from the same vendor, and had been in use for well over a year before we discovered the problem. The problem manifested itself when data packets with all bits on (or nearly all on) were transmitted. The defective repeaters would usually reject the packets and the transmission would eventually time out. This situation - some 500 consecutive bytes all binary ones - is fairly rare, but happened to occur in a new version of a popular application installed on a public access file server. When people were unable to copy the application across the network, the application itself was blamed. As a workaround, it was simply replaced by a self extracting compressed archive, which happened to remove the problematic bit pattern. We contrast below how the problem affected backups, where of course such a workaround is not available.

the computer as scapegoat - Having found one explanation for a failure, it is easy to stop at that point, and it is satisfying to have found a scapegoat. This attitude, however, can be a barrier to accountability because it discourages probing deeper. For example, some people had been experiencing strange problems with recent Mac Quadra 840AV models. Machines would crash or freeze for no apparent reason. Using "composite" memory chips is not recommended - especially for these models, and was cited as a possible explanation, but no one actually looked inside to check. It was simply asserted that composite memory wasn't used. Until this explanation was finally validated, the AV model type was made a scapegoat.

Without accountability, failures can be seen as something mysterious and unknowable, or as unfortunate accidents that cannot have been prevented. Conversely, acting on the assumption that someone is accountable directs attention to solving the problem at hand. Such attention is worthwhile because it is often effective. We have tried to illustrate this below. Reliability is enhanced; and often equally important, so is the perception of reliability.

Operation

Backups are performed by a Unix program, *macdump*, which sends requests over an AppleTalk network to a Mac System Extension, *Dumper*, loaded in the target Mac at boot time. Backups are demand driven and controlled from the Unix end. Files from each Mac are consolidated in one Unix file. A number of backups are performed in parallel and managed by a separate program, *macdumper*. A Unix filesystem is used as a buffer, and the number of concurrent dumps is adjusted according to disk space available. Completed backup files are written to tape, and removed from the disk.

The infrastructure divides naturally into three parts - at the Mac, the network, and the Unix host. The kinds of failures run the gamut from the mundane to the esoteric, as well as those related to timeouts, which are a fact of life on the network.

At the Mac

network identity - AppleTalk uses the concept of Network Visible Entities to identify services. These are (name, type, zone) tuples in a table on each mac. A node (Mac or Unix host) can send a broadcast request on the network to look up a particular tuple. The Mac matching the request returns its network number and node number which is used in all further transactions. macdump looks for a Mac based on "Owner Name" (for the type registered by Dumper, and within a particular zone).

The target Mac won't be found unless the name expected matches character for character with the name as entered by the user. Finding the Mac is the first step. macdump recognizes this situation and sends an explicit messaged alerting the user. For some users, this is enough; others need a number of nagging messages night after night. The situation can become sublte when the user inadvertently enters extra spaces or even invisible characters in the name. The Mac will also be reported "not found" if the Mac is located in a different zone than expected. This can happen due to office moves, network configuration changes, or deliberate user action (with EtherTalk).

disk names - once the Mac has been "found", macdump tries to connect to the disk to initiate backups. This can be frustrated for a number of reasons. A basic problem is that the user has changed the disk name. The same comments apply as for the owner name above.

trackball - some trackballs have a button, which when latched, automatically pulls down menus when the cursor passes over them. The latchd setting somehow prevents Dumper from running, which causes macdump to timeout.

hyperactive screensavers - Screensavers are very popular. Some have parameters which can be set to generate complex patterns, but in so doing don't give Dumper sufficient time to run, which again causes macdump to timeout. Some people run movie loops as screensavers, which also starves Dumper.

competition for network bandwidth - Dumper will also timeout if the target Mac is already too busy generating or receiving network traffic. This can occur, for example, when trying to print to a network printer which is experiencing problems that entail continul retries and error messages. A similar failure occurs if the target Mac is running the QuickMail NameServer while the NameServer is updating its database from other MailCenters.

games - some games effectively take over the whole computer, denying other tasks, such as Dumper, an opportunity to run. In one stubborn case, involving a Mac shared by several students, the game was only run at night for recreation after the daily work was finished. This problem was only detected by probing the Mac every few minutes for potential timeouts and niting which processes were running.

"shutdown" though still powered up - Selecting the Shutdown menu command is designed to gracefully shutdown all processes before powering off. Sometimes users select Shutdown, but still leave the Mac powered on. In this situation, the Mac will still be visible on the network, but unable to respond to backup requests.

Extensions Off - users can elect to disable all System Extensions at boot time. This affects Dumper as well, so the Mac is effectively "not found".

clock setting wrong - Dumper sets the "last backup time" field in the file name table after a successful backup. Most backups sessions are "incremental", rather than a full backup. i.e. only files changed since the last full backup are dumped. If an accurate clock setting is not maintained, this can result in too few or too many files in the incremental dump. Other misleading situations are also possible.

unreadable files - An especially subtle situation can occur when file sharing is enabled. This feature allows a remote Mac to mount one's local disk over the network. It is possible for the remote Mac to open a file with exclusive read access. In fact, some of the most popular productivity software for the Mac does his. This denies access to Dumper as well, so such files cannot be backed up.

cooperative multitasking - Under the Mac multitasking model, foreground programs are expected to periodically release control to the operating system to give other processes (such as Dumper) a chance to run. Some programs, including some of the most common ones, such as the Finder, release control relatively infrequently. This can cause substantial delays in the dialog between macdump and Dumper and cause backups to run several times more slowly.

This can effectively amount to failure when the time to complete a backup is prohibitively long. To work around this problem, we developed an auxiliary program which insists on rapid service from the operating system and in turn gives control more gives control more frequently to Dumper. Subscribers with this problem, which is more acute on certain Mac models than others, are asked to keep this program running all the time.

Unix

prime gateway failure - The Unix software is linked with a library implementing the AppleTalk protocols originally developed at Columbia Univ. (CAP). CAP uses a statically assigned AT-IP gateway for all outgoing packets. Backups will be disrupted if the prime gateway crashes or becomes unreachable. We have modified macdumper to detect this situation and pause till a secondary gateway can be designated. This is done by an independent process which runs as root so that it is able to make the change.

crontab - The backups are started automatically each night by an entry in the Unix crontab file. Occasionally the cron daemon that processes crontab entries will crash. To detect this problem, we have set up other entries so that two hosts will periodically check up on each other to make sure cron is running.

tapes in drive and writeable - Other crontab entries check that fresh tapes have been loaded in the drives for the coming night's run.

router problems - macdump returns only generic error messages such as "Mac not found", "connection rejected", or timeout. Normally they should be taken at face value, but a pattern of failures can indicate router problems instead. Several failures at nearly the same time, for example, are usually caused by a break in network connectivity.

tape problems - Errors when writing to tape are particularly vexing since there doesn't seem to be any way to write over or past the place where the error occurred on the tape (though theoretically this strategy should work). The best workaround we've found is to let the backups currently running continue to completion, and manually initiate a job to write the files to tape in the morning.

Network

multiport repeaters - A particularly mysterious problem involved a number of different Macs which together began to experience timeouts for no apparent reason. Instrumenting the backup process showed that backups were often failing in the midst of dumping the same file name - a Help file for a new version of a popular application. We attempted to reproduce the problem by placing the files on a Mac more accessible to investigation, and in one case by physically moving the affected Mac to a part of the network where it was easier to observe -

but the problem disappeared! Until we were able to reproduce the problem at a place on the network where it was convenient to run a packet level monitor - which revealed the multiport repeater as the problem - it was almost inconceivable that such an integral part of the network could be at fault, especially since it had been in use for so long without reported problems. The specific problem - dropping packets with bits all ones - could have affected anyone copying files, etc. across the network, but would probably never have been diagnosed .

Discussion and Summary

In nearly all the failure modes encountered, we have been able to understand the problem well enough to establish a line of accountability. For example, once we demonstrated the problem with mulitport repeaters, the vendor was forthcoming with replacement models.

It was somewhat surprising to realize how big a role the user has in overall reliability. Most of the failure modes at the Mac are not apparent to the user, yet only the user can take (or reverse) the action needed to fix the problem.

Our strategy, once such a situation is detected, is to hold the user "accountable", though without the pejorative connotation this word has in common usage. The role of the administrator is to understand the failure mode well enough to assign accountability, and then alert the user (in such situations) why the backup failed and what steps are needed to correct the problem. Where possible, users are notified by automatically generated e-mail messages. Repeated failures bring repeated messages, and the cumulative nagging effect is sometimes needed.

Acknowledgements

This work was supported by the Director of the Scientific Computings Staff, Office of Energy Research, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

References

[1] Nissenbaum, H. Computing and Accountability, *Commun. ACM 37, 1* (1994), 73-80.