1 of 1

Conf. 931018--21

UCRL-JC-114230
PREPRINT

Storing Data from Fusion Experiments at the National Storage Laboratory

D.N. Butner and W.H. Meyer

This paper was prepared for submittal to the
15th IEEE Symposium on Fusion Engineering
Hyannis, MA
October 11-15, 1993

September 3, 1993

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

STORING DATA FROM FUSION EXPERIMENTS AT THE NATIONAL STORAGE LABORATORY*

D. N. Butner and W. H. Meyer
Lawrence Livermore National Laboratory
P.O. Box 808, L-636
Livermore, CA 94550

ABSTRACT

The National Storage Laboratory (NSL) at the National Energy Research Supercomputer Center (NERSC) is a prototype facility which is developing data storage and retrieval techniques using hardware that includes a hierarchy of storage devices. The ultimate goal is to store terabytes of data and achieve rapid retrieval times compatible with the type of media where the data is stored.

Files stored in the NSL are accessed directly using the Network File System (NFS); in the future, the Andrew File System (AFS) is expected to be used. System level control of files is available using the File Transfer Protocol (FTP) or a set of program-callable routines. We have experimented with storing and retrieving data from fusion experiments at LLNL and at General Atomics in San Diego, California, using computers running UNIX and VMS operating systems. We discuss some issues associated with accessing files whose names are known, but which are not immediately available, the time required for retrieval, and other pertinent parameters.

INTRODUCTION

Two aspects of magnetic fusion experiments have led us to look at the National Storage Laboratory as a prototype for the storage of data. First, existing experiments generate ever increasing amounts of data. For example, the DIII-D tokamak experiment at General Atomics in San Diego is generating 30 to 40 megabytes of compressed data per shot. It is anticipated that the TPX experiment will generate a few hundred gigabytes of compressed data per day[1]. Second, magnetic fusion experiments are becoming national and international in scope. One possibility for the management of this large amount of shared data is the use of a facility similar to the NSL.

The NSL development has several goals, such as direct connection of storage devices to networks for high speed transfer and multiple, dynamic, distributed storage hierarchies[2]. We are primarily interested in the network accessible storage hierarchy concept. The important concept is that the data files are stored in a normal UNIX directory structure, defined by the software developers associated with the experiment. The storage facility would maintain the information about the physical location of the data; programs would access the data independently of the physical location, using only a UNIX pathname.

Fig. 1 An example of a complex hierarchy

The NSL potentially has a complex hierarchy of storage devices as shown in Fig. 1. Incoming files would be stored on solid state disk, later migrating to disk, tape, and the remote library; when a file that is no longer on disk is accessed, it would be automatically retrieved from the remote library or tape. We are currently using only the part that migrates disk data to 8mm tape.

EXPERIMENTS WITH THE NSL

We have been testing various methods of interacting with the NSL in order to understand the issues that may arise in the future use of such facilities. We have accessed the NSL using FTP and NFS from both UNIX and VAX/VMS computers. Each access method has tradeoffs that must be considered. The FTP protocol is useful when the a complete file must be moved into or out of the NSL, and with FTP, a user or program can determine whether a file is currently stored in the NSL disk cache or not. Due to its ability to perform random accesses to files, the NFS protocol is useful once a file is stored in the NSL and a user or program needs to access only a portion of the file.

MASTER

In either case, problems can arise if a file is not present in the disk cache of the NSL. The operating systems and existing processes that we have used expect that if there is a file system directory entry for a file then the file is immediately available on disk; any error that occurs in accessing the file is fatal, and no retries are attempted. This expectation is not appropriate for the NSL, in which a file system directory entry will always exist, but the file may not be immediately available.

When the NFS protocol is used, the remote file system (the NSL) may be mounted either with a hard connection or a soft connection. When the hard connection is used, the local process will wait until the file is retrieved from secondary storage and placed on disk at the NSL. The problem with the hard connection is that most processes are written so that they just wait (since local data is normally immediately available) without being able to report to the user that the file is being retrieved. The time to retrieve the file can vary significantly, depending on its migration location and the number of files being retrieved concurrently. The user may give up and interrupt the process; when the user tries again, the file may or may not be immediately accessed. A user who does not realize that there is a hierarchical file system holding the files may become frustrated at the unpredictable response.

With a soft connection, an error will be reported by the local operating system after a predetermined time-out period. Existing processes often treat such an error as fatal and give up. The user may try again in a short time and the file may or may not be immediately accessed. The user may be frustrated by this behavior also.

With FTP, when a user requests a directory listing, the FTP server at the NSL responds with more information than typical servers. Included in the response is an indication of the current location of the file, disk or tape. When a user asks to copy a file that is not on disk, the FTP server reports back that the file is being restored from tape; however, the user must keep trying until the file is restored.

When the NSL is accessed from VAX/VMS systems, additional problems arise. We have tested only the Multinet TCP/IP package with VMS, but we would expect that other packages would have similar problems.

One problem arises because VMS has several file types, whereas UNIX (and hence the NSL) has only one file type: a stream file. When a file is copied from a VMS system to a UNIX system using the NFS protocol, Multinet handles this dichotomy by creating an "invisible" file on the UNIX system as shown in Fig. 2. The filename starts with the string ".*\$fdl*"; this file contains an ASCII description of the file in the VMS File Descriptor Language (FDL). The bytes from the VMS system are then stored appropriately on the UNIX system. Note that processes on the UNIX system cannot read the file properly unless they can parse the FDL file. When the file is

Fig. 2 File and file descriptor locations

read from a VMS system using the NFS protocol, Multinet first reads the FDL file in order to determine the file parameters on the VMS system; it then reads the file and interprets the data properly for VMS. No FDL file is created on the UNIX computer if the VMS file type is *stream_if*, the only file type available on UNIX.

As with other VMS processes, Multinet expects that the FDL file will be present on the UNIX system if a directory entry exists for it. If there is an FDL file but it has been deleted from the disk cache, as shown for *vmsfile2* in Fig. 2, an error is returned to Multinet and it assumes that the file type is *stream_if*. In addition, the file type information is cached in memory at the local computer, so even when the FDL file is retrieved at the NSL, the local computer continues to treat the file type as *stream_if*.

The FTP protocol does not use the FDL file; therefore, files with VMS-specific formats cannot be successfully stored and retrieved using the NSL.

POSSIBLE PROBLEM SOLUTIONS

This problem with file types basically occurs because of the use of heterogeneous computers and operating systems. A general solution to the problem is to use stream files and to require the application programs to interpret the file information. The files generated at DIII-D can be handled in this manner. In general, the use of self-describing files may be a solution to this problem. An example of such files are the Hierarchical Data Format (HDF) files defined by the National Center for Supercomputing Applications (NCSA)[3]

There are several possible solutions to the file access problem.

We tested one solution in conjunction with the data management system for the Microwave Tokamak Experiment (MTX) at LLNL[4]. In this system, if a user determines that a shot data file is not available on local disk, the user requests

that the data file be retrieved from the local tape archive. A set of processes retrieves the file and notifies the user when the file is restored to disk. An additional process was added to this system; before an attempt is made to retrieve the file from local tape, this process determines if the file is stored at the NSL. If so, the process accesses the file at the NSL to force a retrieval and then periodically checks to see if the file has been retrieved. When the file is retrieved, the user is notified. At the time this process was added, the FTP protocol was not available, so the operation depends on an NFS soft mount. This process could now be implemented to use FTP to determine whether the file is on disk or not, and if it is not on disk, to also use FTP to force a restore.

Other possible solutions (which we have not tested) involve modification of the processes that access the files. For example, the processes could access the files using an NSL server process on the local computer; this process would understand the idiosyncrasies of the NSL and notify the accessing processes of the status of files. Each accessing process could also be modified to handle the new types of status conditions returned by the NSL.

TIMING MEASUREMENTS

The NFS and FTP access to the NSL provide transfer rates that are generally typical for these protocols, 10 to 100 kilobytes per second. These numbers are limited by the 10 megabit per second ethernet connections that we are using. The higher speed networks that will be available in the future will be required to handle the volume of data generated by the TPX experiment.

The retrieval of files from 8mm tape to disk occur in from 1 to 2 minutes under the best conditions. If we concurrently request 50 files which were stored at the NSL at various times, which therefore are probably not stored adjacently on tape, file retrieval can require 5 to 10 minutes. These times are much better than our local retrieval from 8mm magnetic tape, which can take from 10 minutes to several hours.

SUMMARY

The National Storage Laboratory is providing a platform on which to test methods of storing and accessing large amounts of data in a networked hierarchical storage system. Our testing shows that such storage systems will be useful, but that use of the hierarchical system must be fully incorporated into the system design.

ACKNOWLEDGMENT

*Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract number W-7405-Eng-48

REFERENCES

- [1] R.A. Coyne, H. Hulen, and R. Watson, "Storage Systems for National Information Assets", *Supercomputing '92*, November, 1992
- [2] S. Davis, P. Funk, P. Hagar, G. Oliaro, W. Rauch, W. Stark, D. Butner, and T. Casper, "Conceptual Design for the TPX Central Instrumentation and Control System", presented at *15th Symposium on Fusion Engineering*, Hyannis, MA., October 1993.
- [3] Data Link, National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign, January/February, 1989.
- [4] D.N. Butner, M.D. Brown, T.A. Casper, W.H. Meyer, and J.M. Moller, "Diagnostics Data Management on MTX", *Proceedings 14th Symposium on Fusion Engineering*, San Diego, CA, October 1991, pp. 847-851.

**DATE
FILMED**

1 / 3 / 94

END