Pilot Deployment of the ITC File System

John H Howard
September 28, 1984

Introduction

Our grand plan for developing the ITC file system calls for an initial version which establishes architectural directions and gives us something concrete to measure and learn from, followed by a re-evaluation, redesign, and re-implementation resulting in a subsystem with sufficient speed and operability to support the entire campus. Given that we have such an initial version in hand, it seems reasonable that we should include it in the planned pilot deployment of some fifty interim workstations to selected application developers on campus.

I see several benefits from such a deployment. It will force us to confront the inevitable problems and oversights we might otherwise put off indefinitely. While this may be unpleasant initially, it will result in much greater confidence on our part that we have done a thorough job. It will provide a credible source of live workload data which we can use both for identifying problems in our current design and for estimating critical system parameters such as cluster size. It will make software distribution much easier than using some sort of remote copy, since we can simply store the current version of our stuff in VICE and expect it to be available immediately to cooperative users. Last and perhaps most important, a pilot deployment will be a natural dividing line between the initial version and the intended redesign; having carried it off we will be able to sit back and think for a while rather than continuing indefinitely to patch up Version Zero.

Of course, we must expect some costs. We really will have to deal with the oversights and spend enough on performance to make the deployed system tolerable to use. We'll have to do some documentation and training, with which some of us may feel uncomfortable, and if we do a poor job of it we may be faced with a continuing job of hand-holding.

Therefore I am soliciting from the ITC, and in particular the file system group, ideas for remedying deficiencies of the current file system and for makeing it easy to deploy, maintain, and use. This document summarizes both my own ideas and your inputs to date.

Proposed Pilot System

The machines we deploy will have local disks containing (small) root file systems, just enough to bootstrap the machine, and connections to the VICE file servers for the bulk of our deployed programs and system libraries. Although it will be possible for users to store their files directly on the local disks, we will recommend that they use VICE, citing both the sharing capabilities that provides and the fact that we will make backup copies of the VICE data but not of the local disks.

There will actually be two independent VICE file systems; one for general use (both external and within the ITC) and one for ITC use only. The main reason for this division is to reduce concerns about our file security controls. The ITC subsystem will probably be named "/itc" and contain ITC home directories, source code, and other sensitive information. The external file subsystem will be "/cmu" and will contain both external users' directories and deployment copies of our system software.

Pending resolution of current functional and performance problems with VICE's automatic mechanisms for making backup copies of files between servers, we will turn the backup system (but not replication of system files) off. Instead we will arrange for at least one machine with a good tape capability to connect to VICE as a workstation and make incremental tape backups on a nightly basis. This will require installation of VENUS (and the supporting kernel modifications) on a Vax, since there appears to be no adequate tape for Suns. We recommend ordering another VAX 750 (with 4MB of memory, tape, and two disks) as a server for the /cmu file system, both to provide the tape backup capability and to serve as insurance against continued unreliability in the SUN 170's.

We will modify the Unix *login* program to try looking in a password file stored in VICE first, before checking the one on the local file system. The local password file should be retained to provide for independent operation, and *login* should replicate entries from the global password file into the local one as they are (successfully) used. At some future date we might write a program to remove local entries after verifying that their user ID numbers are not referred to in the local file system or cache (this takes an inode scan). With less than a hundred users for the next year, this should not be immediately pressing. Perhaps more important will be some automatic way to maintain user registry data, which in the current system is distributed between the password file, the authentication server, and the protection server.

It has been proposed that we provide a disk server in addition to the file server and local disks. This is justified by the feeling that it will provide a large shared (but read-only) pool of data and programs without the performance exposures of VICE. We recommend *against* this proposal on the grounds that it will be an additional source of failures. Our experience is that one can load binaries from VICE with only a small performance impact. We don't yet have sufficient information about other shared information (libraries), however. Performance-critical files which are essentially read-only (to the workstation) could be copied from VICE to the local disk by a cron-file entry or as part of logging in.

Work Items

1. Modify *login* to use a global password file, to perform the initial handshake with the authentication server, and to leave the session password where *venus* and other system components can find it. Preferably it should cache entries in the local /etc/passwd; initially we may simply copy the entire global file.

2. Complete preliminary performance measurements and identify necessary server modifications.

3. Modify the existing file server code as needed above, for example by

suppressing the user file backup mechanism and providing a tape backup procedure instead.

4. Unify our various Unix kernel sources to the point that it is possible to regenerate existing kernels from the unified source.

5. Migrate the *virtue* file system intercept in the kernel to Sun 120's, Sun 170's, and Vax 750.

6. Migrate *venus* itself to all of the above. Add modifications as required.

7. Define and document procedures for installing new file servers, new workstations, and new users. Establish backup and other operational procedures, including logging and reporting both hardware and software problems.

8. Order and install a second Vax 750 and bring up the /cmu file subsystem on it. Install and configure additional server machines (Sun 170's) as the number of deployed workstations grows.

/vice/itc/jhh/memos/plans/fspilot.d