Dropping the Mainframe Without Crushing the Users: Mainframe to Distributed UNIX in Nine Months

Peter Van Epp & Bill Baines – Simon Fraser University

ABSTRACT

This paper describes the first phase of the evolution of the computing environment at Simon Fraser University from a centralized, long established, mainframe based computing environment to a distributed UNIX based computing environment.

The first section of the paper gives a brief history of the site and some of the previous initiatives towards distributed computing.

The second section describes the Task Force report that mandated a conversion to UNIX within 9 months and the plan that was developed to do that.

The third section details the migration plan versus migration reality.

We then extract from this experience the major issues encountered in converting from a mainframe environment to UNIX. We also mention some things that are taken for granted in the mainframe world that don't seem to exist in the UNIX world. This should be of interest to sites planning to convert from a mainframe to UNIX, and to UNIX vendors that are interested in selling machines in that market. We also describe how we resolved some of the problem areas.

Next we describe what we did right, what we did wrong, and what we would repeat if given the opportunity. We also describe some UNIX experiences encountered when pushing the 'shrink wrap' envelope with an implementation that started out with 18,000 user accounts.

The last sections of the paper detail some of the new work we are undertaking to resolve existing problems, and add additional function. Our conclusions attempt to summarize our major points.

History of the Site

Since the mid 1970s central academic computing at SFU was done on a series of IBM mainframes running the Michigan Terminal System (MTS). MTS was initiated at the University of Michigan and was enhanced and maintained by UM and the 7 other universities that run it. Since MTS was written specifically for the academic time sharing environment it evolved over the years into a very secure system, (as the students found security holes they were plugged) with a very rich set of access control mechanisms. There are however, several problems with the model. With no manufacturer supporting the operating system, each site was required to have staff capable of doing operating system kernel level development and support. Staff at this level are both hard to find and hard to keep, and since there are so few sites using the system, training them was a large burden. The mainframe that MTS runs on is expensive in both capital cost and maintenance. There are many things that a mainframe does very well (high volume, high speed I/O comes to mind), but equally many things that a mainframe does poorly (interactive graphics support, character by character I/O).

In the early 1980s, the MTS community recognized that the computing world was changing and that distributed computing was the wave of the future. There were discussions about the activities that the various sites would undertake to move to a distributed environment and what role, if any, MTS would play in that environment. One thing that became apparent was that the "dumb" terminal over telephone wire to the central machine was going to give way to a network of machines distributed across many different areas. As a result of this realization, and concurrent with a major telephone system upgrade at SFU, the computing department designed and installed the physical portion of a new high speed network. At that time (1985), it was not clear whether Ethernet or Token Ring would ultimately dominate, and FDDI was on the horizon. The decision was made to create two major nodes each connected to the other by multi-mode fibre optic cable that would hopefully support FDDI when it became available. These two network centers were connected to wiring closets (initially 38 and now up to 50 and climbing) via multi mode fibre. The wiring closets were connected via shielded twisted pair cable to every place where there was a phone installed. The cable could support either token ring or Ethernet connections with equipment available at that time, and now of course, supports 10BaseT connections and looks like it will be able to support twisted pair FDDI when that becomes available.

With this infrastructure in place we then proceeded (as funding permitted) to connect the various Macs and PCs around campus into departmental Lans (using the new high speed network wiring) and allow them to access the mainframe, (through a new PDP11 based interface that supported TCP/IP and TN3270 into the IBM mainframe and therefore MTS) and the Internet, either directly or via logging on to MTS and then going out to the Internet.

As the start of the phaseout of MTS, we identified specific functions that could be better done in a distributed manner and started implementing them with the vision that at some point in the future, there would be no more user functions on MTS. MTS would either operate as a "server" for various functions, or be replaced by some better and cheaper system – with minimum user impact. The first function to be replaced was text processing. The norm at that time was an nroff like package (called Textform) that ran on MTS and the standard MTS text editor. Both of these were clearly inferior to the packages available on the Macs and PCs, and we therefore installed 75 IBM PS2 model 25's and 75 Mac Plus's with Microsoft Word, one dot matrix draft printer for each 3 machines and 2 LaserWriters for final output. This facility was in the University library, and was named 'The WordStation'. It quickly became the place of choice for all types of word processing. Support for Textform text processing was removed from MTS.

The next easily identifiable group of users was the Numerically Intensive Computing users. At that time they were using all the cycles that they could get on the mainframe, but since they were sharing it with up to 200 time share users, they obtained fewer CPU cycles than they wanted and they impacted response for the other users. Their jobs were CPU bound, did almost no I/O and therefore they didn't benefit from the I/O performance of the mainframe. There were only 8 to 10 users in this category, and they were moved to a pair of 4 processor Silicon Graphics (SGI-240) UNIX machines. Today, this SGI complex has 20 processors across three systems and is still growing. At the time of the switch over, there were reports that the users were seeing as much as an 8 times increase in performance on their jobs. They were no longer competing with the 200 time share users, the SGI CPUs were each more powerful than the IBM mainframe in CPU performance.

These two steps reduced the load on the mainframe enough that a new machine did not have to be purchased. The next step was to size up an instructional microcomputer facility. A questionnaire was sent to all faculty asking what kind of microcomputers, what software and how many hours per week they needed in support of courses that could use computing. The response prompted us to purchase a lab of 75 IBM PS2 386sx PCs. These machines were connected to the high speed network to allow connections to the mainframe and the Internet, and today, are supported by three Novell servers for file services and software. In addition, a reservation program running on Novell was written that allows an instructor to book a certain number of hours per week per student for a course. The software then allows the student to book a machine in advance for a specific time period to do his or her course work, at times when the machine is not booked in this way it may be used on a 'walk in' basis for course work. This facility reduced the load on the mainframe even further. Today it is overbooked by almost 200% and still is not used to its full capacity (i.e. the faculty are over estimating the amount of computer time the students need, some students have their own PCs and some students drop out).

This was basically the state of the world at SFU on Feb. 6, 1991 when the report from the task force on computing was released, and computing at SFU was once again drastically changed.

The Task Force Report

In August of 1990, the computing center stopped reporting to the Vice President for Research and started reporting to the Associate Vice President Academic. The new Vice President commissioned a task force of senior academics to examine and make recommendations on computing at SFU. On Feb. 6, 1991, the task force released its report, and the Associate Vice President acted upon the recommendations.

The recommendations included:

1. UNIX is the academic operating system of choice, and SFU should move to it.
2. By August 31, 1991, the electronic mail system will be moved to a platform other than MTS.
3. By December 31, 1991, all computing will be moved to UNIX and MTS will be shut down.
4. The IBM 3081 mainframe should be decommissioned on Dec. 31, 1991.
5. The direction of computing at SFU will be in the hands of a hierarchy of committees composed of representatives from the user community, and these committees will set computing policy and direction for the computing center.
6. The computing center will be split into two new departments,
   - 6a) Academic Computing Services – responsible for the implementation and

support of all academic computing.

6b) Operations and Technical Support – responsible for the day to day operation of all the central site machines, both academic and administrative.

The staff of the computing center that had been supporting the administrative systems were distributed to the various administrative departments, and those departments became responsible for the maintenance and support of their own applications.

The staff of the operations group was not significantly impacted.

The recommended time table implied that we had to convert from a mainframe environment with very little UNIX experience to a production UNIX environment within 9 months. UNIX support in the computing center at that time consisted of a single UNIX consultant that we had hired away from our School of Computing Science. He was supporting a departmental Sun 4/280 as a pilot UNIX service for both the department and the community, and the Numerical Computing SGI complex. Very few other people in the department of 60 people that remained after the split up had ever even used UNIX and none of those except for the lone UNIX consultant had ever done any serious UNIX system administration.

The Plan

Once the shock had died down a little (one consequence of the committees taking over the policy making role was 6 of the computing center managers were made redundant and no longer had jobs), we began planning how we were going to get from where we where to UNIX, and how "UNIX" would look. For the first several months, there were meetings pretty much every day: first to identify MTS functionality, and then to target replacement services in the brave new world of UNIX.

The committees proposed by the task force report were commissioned, and they started to meet regularly to discuss how the changes would be made. Some committees solicited input from the user community.

One of the first decisions made was that the new UNIX system would be a distributed system, and therefore there would not be a mainframe-like UNIX host, but rather a series of smaller UNIX hosts. Given the shortness of the time frames involved, it was further decided that the conversion would be done in two stages. The first stage would be to replace exactly the functionality of the current mainframe system with some number of UNIX systems that would reside in the central machine room (this to be done by the deadlines specified in the task force report) followed by whatever further distribution the community saw fit once some experience with UNIX was gained by both the users and the computing staff.

The gathering of the research community's application requirements and the list of replacement applications for existing MTS services was distilled down to a list of what packages were available on UNIX (with costs) and a list of applications that might have to be dropped. From this, a manageable list of what we proposed to support was formulated. The committees' initial cut on a configuration was to buy a server for each of the 37 departments at SFU, but when the cost of software (and the fact that the software costs were per machine) was compared to the budget, it was quickly decided that 5 servers, one for each faculty, was more in line with what could be afforded. Several of the schools realized that their software requirements were similar, so they merged their servers. The final plan included three research servers, divided along the lines of function; a database server, a statistics server, and a compute server. Access to these machines would be restricted to faculty and graduate students. The instructional committee chose two machines for instructional use, and a general login server for e-mail and conferencing was specified. The initial implementation total would be six hosts (3 research, 2 instructional, and 1 general purpose).

The committees were surprised at the true cost of distributed computing. During their early consultations they discovered that the cost of multiple copies of software across many UNIX hosts could easily exceed the prices being paid for mainframe software. Some software vendors even classed new RISC UNIX machines as supercomputers, and charged appropriately. The reduced initial capital costs of the UNIX hardware did not fully offset the surprisingly high software costs.

In parallel with the committee effort, we were profiling the current usage of MTS, this necessitated making modifications to MTS to collect the data. The data collection ran for a few months with the following results:

E-mail and Conferencing	28%
Statistical packages	20%
Compilers	15%
Custom Applications	18%
Symbolic Mathematics	4%
Numerical Analysis	4%
Text Processing	4%
Utilities	2%
Databases	3%
Graphics	2%

We identified every function we could find that was being used on MTS, and summarized the results in a document. We then identified adequate UNIX based replacements, and those that had less than adequate UNIX based replacements. A few functions had to be dropped.

At the same time, another group of the computing center staff were examining the issues of providing a UNIX service and deciding on a recommended architecture for the UNIX system to be submitted to the committees for approval. The two tasks, detailing what we had now, and deciding on what the UNIX system should look like, of course interact to a large extent, with one influencing the other, and so there was a lot of intergroup discussion.

The broad outlines of what we thought the UNIX service should look like follows:

- All people that are members of the university community should have automatic access to a UNIX account for free. This was largely true of the MTS system, with undergrad students being eligible for a resource limited account on MTS on request.
- The model of MIT's Project Athena for access should be followed if possible (i.e., the Athena motto "wherever you go there you are") so that the users home directory and environment should be the same no matter what machine he or she is logged on to.
- In support of the last statement about access, and to continue the mainframe concept of backup of user files on a regular basis at an affordable cost, there should be a single central file server that holds all of the users files and can serve them over the network to whatever machine the user is logged in to.

This all having been worked out, we then turned our attention to how all of this would be implemented. This was going to be a major and disruptive change for everyone involved especially the researchers who had been using MTS for many many years and had programs written to take advantage of packages and features on MTS that wouldn't be available on UNIX. In order to give them time to move, it was desirable to have the UNIX system up by July 31, so they would have all of summer semester to move their code over to UNIX. Starting with the fall semester (starting in September) all student computing would be moved off of MTS (if possible) leaving us with a fallback to MTS for the fall semester if problems occurred, allowing an orderly shutdown in December.

The Reality

As always the plan failed to survive its first brush with reality. We had thought that we were treading down a well traveled road from a mainframe to UNIX. We discovered that this isn't true. We were not able to find a site (nor were the various UNIX workstation vendors) running large numbers of users on one or more server class machines. It is possible that the exclusion of the mainframe like UNIX vendors was the cause of this, if you have a mainframe, and are moving to UNIX maybe you choose a large UNIX box and move to that.

Restricted Main Memory Bandwidth

The major challenge in moving from a mainframe to UNIX servers appears to be providing sufficient I/O bus and main memory bandwidth on the UNIX server at a cost that doesn't approach that of a mainframe. The UNIX vendors keep pumping up the CPU power of their boxes, but have not been matching the CPU power with an equivalent increase in main memory bandwidth, possibly because doing so isn't cheap.

In order to understand why this is so, a little bit of background on main memory is required. The typical workstation uses Dynamic Random Access Memory (DRAM) as their main memory. DRAMs have typically been doubling in size every couple of years while falling in cost, making them an inexpensive main memory. The access speed of DRAMs, however has not been falling at any where near the rate of the size increase. Due to packaging constraints (pins on the package) and chip architecture (buffers are expensive in chip real estate) the trend has been toward ever larger numbers of bits in a single package (1 megabyte SIMMs, 4 megabyte SIMMs, soon 16 Megabyte SIMMs). This has a rather large performance impact, since typically only 1 byte out of those x million bytes in the SIMM can be accessed at a time, and the cycle time to get to the next one can be several hundred nanoseconds. Note that the cycle time of a DRAM is different (and longer) than the often quoted access time. A DRAM with an access time of 70 nanoseconds has a cycle time of up to several hundred nano seconds.

If we assume that the typical SIMM can be cycled in 200 nanoseconds, then the SIMM has a memory bandwidth of 5 megabytes per second. If we assume a 4 byte wide memory bus, then this gives us a total main memory bandwidth of 20 megabytes per second. For comparison, a typical mainframe has a main memory bandwidth that is between 200 and 800 megabytes per second, obtained by using static ram in place of the DRAM, and using memory interleaving to obtain even more speed (but at vastly higher cost).

The next thing to consider is that this 20 Megabyte per second bandwidth into memory has to be shared between the RISC CPU, and the I/O subsystem that is transferring data to and from the various peripherals. The RISC CPU executes (on average) one 32 bit instruction per clock cycle and clock speeds of 25 to 50 Megahertz are not uncommon. If not for instruction and data caches, this would consume all of the available main memory bandwidth feeding the CPU, leaving no main memory bandwidth for doing I/O. If we assume that the caches are such that half the main memory bandwidth is available for I/O, we are left with around 10 megabytes per second to service the I/O requirements of all of the peripherals.

Restricted I/O Paths to Peripherals

Now let's look at the I/O subsystem of a typical IBM mainframe, as noted above, a fancy (and expensive!) main memory subsystem provides an order of magnitude higher main memory bandwidth (even if the cost is probably 2 orders of magnitude larger in the mainframe case). This leaves a considerable amount of bandwidth for the use of the I/O subsystem. The I/O subsystem consists of its own processor (typically called a channel director) that controls the transfer of bytes to or from main memory and the peripherals. The channel director has between 16 and 256 channels (on old to modern mainframes). Each channel can transfer data to a peripheral at 3 megabytes per second, and the main memory bandwidth is such that many (but perhaps not all) of these channels can be transferring data at the same time without causing instruction starvation for the CPU which is also contending for main memory.

Think of an IBM channel as being similar to a SCSI controller on a UNIX box. To replace our mainframe which had 4 channels out to disk, we would need a UNIX box with 4 SCSI controllers each controlling a string of disks, and all capable of being active at the same time. In addition, there were another 3 channels out to high performance tape drives that are not present on the UNIX machines and that I/O load has now moved to the disks as well.

If all 4 of those channels transfer data at the same time, we would need a path from the disks into main memory on the UNIX box of 12 megabytes per second, but the whole main memory bandwidth is only 20 megabytes per second, and there are still the other peripherals (such as the Ethernet interface) that need some of that I/O bandwidth. Our smallish mainframe is already taxing the I/O capacity of a single UNIX box. As you can see, a larger mainframe site with more channels to disk would quickly overwhelm a single UNIX box.

The obvious answer to this problem is to spread the I/O over multiple UNIX boxes (which is what we did). The challenge becomes to do this in such a way that no single path in the system becomes a choke point to the I/O flow.

Establishing a Distributed File System

There are two major choices for a central UNIX file server, NFS or AFS. We chose NFS not because we think it is better, but because it is already supported by most UNIX vendors, and our single UNIX person had experience with NFS but not with AFS. Given how little UNIX experience we had, the custom nature of AFS, and the tight time frames, We believe this was the correct choice. NFS expertise can be purchased since there is a lot of NFS around; we expect that AFS expertise is harder to find. That said, AFS is certainly a viable solution, and may well be where we end up in a few years. The advantage to AFS is the reduction in load on the backbone network due to local caching. In our current configuration this is not a problem, if we move towards placing workstations on people's desks supported by the central file server (e.g., to provide cost effective backup services) then this will become an issue. One of our fellow MTS sites, Rensselaer Polytechnic Institute (RPI) has chosen to go with AFS to support their cluster of some 400 UNIX workstation seats (some are X terminals), and they have successfully implemented it. The decision will probably hinge on how distributed a site is initially going to be, what networking is in place and how much UNIX expertise is in place.

That decided, we started talking to the various file server vendors about I/O performance on NFS servers. At that time, we found there were few choices (several vendors had just released products, but had neither performance numbers nor a track record), and there little understanding of the issues involved (with the notable exception of Auspex).

The issue here is server performance. We had the choice of buying several (probably 4, one for each disk channel) NFS servers, and then cross mounting them all onto the server machines, or buying a single large server and allowing it to serve all the machines with no cross mounts. If we were to use the single large server, then we are right back to the main memory bandwidth and I/O performance limits of a UNIX box. In order to get the necessary bandwidth to disk, we had already decided that each server machine would have 2 Ethernet cards, one to connect to the backbone and accept telnet connections from users and one dedicated to NFS traffic. This meant that the server had to be able to support between 4 and 6 Ethernet ports and (at this point) 10 gigabytes of disk all fighting for that same main memory bandwidth. In addition the CPU would have to be fielding all of the interrupts and doing the Ethernet and NFS processing for all this I/O. The Auspex solution of dedicating a CPU to each function (one CPU to each two Ethernet ports, 2 disks per SCSI controller all tied together by a very high speed bus) made more sense to us than any alternative we saw.

We ordered an Auspex NS5000 NFS file server with 10 gigabytes of disk and 6 Ethernets to spread the I/O load around to the 6 server machines. We ordered 4 Silicon Graphics 4D320 machines (2 processor 33 mhz R3000 CPUs) for the three research machines and the general login server, and 2 Sun 470s for the instructional machines. As noted above, each of these machines has a second Ethernet card to connect to one of the 6 Ethernet ports on the Auspex file server to provide NFS service in a way that is both secure (because of the private net) and does not add load to the campus backbone.

Analyzing the Computing Load

The choices of machines were arrived at like this: around 30% of the MTS load of 180 simultaneous logins was taken up by E-mail and conferencing, NetNews access (which didn't exist on MTS) was expected to add some more load, so the general login server was allocated 80 out of those 180 users (40%), but they would all be light duty. The statistical users were another 20% and were basically divided evenly between two of the research machines at 20 users each the third research machine was set to be also 20 users. The remaining 40 users were assumed to be students and split 20 users each to the two instructional machines. The remaining 20 people were considered taken up by the RS6000/530 that takes intermediate CPU users (defined as needing between 500 and 1000 CPU hours per year, as opposed to large scale users whose CPU demand is essentially infinite!).

Three Sun Sparc2 machines provide infrastructure machines, one for NIS, NetNews, and syslog; one for DNS, X.500, and the mail hub, and the third as a PostScript converter and driver for our Xerox 4090 laser printer. A VAX/VMS system was reinstalled to run several services that seemed to be easier to implement under VMS.

The 180 user number used above was arrived at because that was the typical load on the MTS machine at the end, and the initial commitment was to provide exactly the same level of service on the UNIX hosts, no more, no less. This was to be true because the UNIX conversion was being funded with the same money currently supporting the mainframe (which as you can imagine caused interesting problems during the time when both sets of machines existed and were being paid for with the same money ...), and any enhancements had to be approved by the committees with the indication of where funding would be found to pay for the enhancement.

Unfortunately, when these machines were ordered, the earliest delivery we could get was mid August – there went the "convert the researchers over the summer" theory, the machines would not arrive in time.

At about this point, we realized that in order for E-mail to be cut over about 90% of the work of the conversion was going to have to be done (i.e., being able to get mail on the UNIX box implied that you be able to log on to the UNIX box, which in turn implies that you have an account and a home directory on the file server etc.) This started a process of talking both the Registrar's Office and the Personnel Department out of the data for students (from the registrar) and faculty and staff (from personnel) so that we could create accounts for everybody. The file server and the SGI CPUs arrived the same day (Aug 26) and both were up within a couple of days (fine support from both companies!) and the mad race to get the systems configured and some 20,000 accounts and home directories set up and installed was underway.

In actual fact, the E-mail cutover happened on Sept 12 (12 days late) after a huge effort (including one 72 hour straight session for two of the people babysitting the account / home directory creation task). This turned out to be too late for the fall semester student computing load too, so it went ahead on MTS. The next several months saw additional software packages and any number of problems found and fixed, and in fact MTS was shut off to user access on January 2, 1992 right on schedule. Delay in the installation of terminal servers to replace the PDP11s meant that MTS actually was still up until March 13, 1992. This turned out well, since many people didn't believe MTS would actually go, and therefore hadn't moved any of their files. MTS being up allowed us to FTP the files rather than attempt to recover them from tape. All in all, this conversion went far more smoothly (and in a far more timely fashion) than anyone believed possible.

Major Issues Raised by the Conversion from a Mainframe to UNIX

Many issues confronted us in the conversion from a long entrenched mainframe system to UNIX, the next section will discuss how they were resolved.

- I/O bandwidth: the IBM 3081 mainframe has 16 I/O channels, each in theory capable of transferring data at 3 megabytes per second and enough main memory bandwidth to allow most of those 16 channels to be transferring data at full speed. Monitoring tools in MTS indicated that the MTS system was as often I/O bound as it was CPU bound with the normal 180 or so users on the system. There is no particular reason to believe that the same would not be true on a UNIX machine supporting the same workload.
- Politics indicated that we couldn't replace the mainframe with a mainframe like UNIX system, indicating the I/O rate had to be distributed across a number of smaller machines (and therefore be somewhat balanced!).
- Magnetic tape support. The mainframe supports both 6250 bpi 12 inch "round" tape drives (IBM 3420s) and IBM 3480 "square" cartridge tapes, with a high bandwidth path (two IBM channels) into the machine. It is very common for a user with a large data set to mount the tape and process the data on the mainframe directly off the tape without copying it to disk first. MTS also can read both all types of IBM standard labeled tapes, ANSI labeled tapes and unlabeled tapes.
- The high speed tape drives make system

backup a not too time consuming job. In the MTS case (as in the safe UNIX case), the system is brought down to single user mode for weekly full backup. This involves about 3 hours to back up some 15 gigabytes of disk to 3480-type tapes.

- Tape library services: MTS provided support for access control of labeled tapes. The label of the tape contained both a tape serial number and the MTS id of the tape's owner. When a tape is mounted, the MTS tape software checked the MTS account name mounting the tape against an access control list before allowing the mount. This is common on mainframe systems, as is having a tape library system to control access to the tapes. At 10,000 tapes we have a smallish tape library, libraries of 100,000 tapes are not uncommon in commercial shops.
- The PDP11 front end system provides an interface to both the SFU library computer and the public switches X.25 service (Datapac in Canada). Since there is a hard dollar cost associated with this, the service needs to be accounted for by user.
- High speed printing: There is a Xerox 4090 92 page per minute laser printer attached to MTS and driven from an IBM channel for campus printing needs. It was printing around 600,000 pages a month when MTS was in use (split between MTS printing and printing from the administrative systems).
- Accounting: the MTS system accounts for CPU time, disk space usage Virtual Memory usage while the job runs, network connect time, printer usage, and also provides limits on all of these resources by user account.
- As pointed out earlier, MTS is a very secure system, offering access protection by access control lists down to the level of a single account (or exclusion down to a single account) for read, write or various kinds of execution (i.e. it is possible to restrict what programs a user can execute against a data file by user id). All of these protections support full wild carding of parameters.
- User directory: MTS implements a directory of all users on the system, which can be searched (via a soundex algorithm) to find a persons E-mail address from an approximation of their name.
- E-mail features: the E-mail system on MTS provides a rich variety of archiving and filtering mechanisms for incoming E-mail.
- Conferencing: a very popular custom conferencing system (called forum) had been written on MTS. A replacement system needed to be found for UNIX.
- A replacement for the PDP11 based custom terminal interface to the mainframe needed to be found.
- Education: both the computer center staff and the user community need to be trained in UNIX.
- In order to create the necessary 18,000-20,000 UNIX accounts that would be required to give everyone on campus a UNIX account, an automated method of account creation would have to be found. There were around 11,000 active accounts (i.e., ones that had accessed files within the last year on MTS, at the point of the switchover).

These issues were resolved in many different ways:

- All users' home directories (currently some 25,000 accounts, some 11,000 of which are active) and any data needed on more than one machine, is stored on an Auspex NS5000 NFS file server. As noted above, this machine has 6 Ethernet ports and started out with 10 gigabytes of disk, and is now up to 18 gigabytes. One Ethernet port connects to the campus backbone, and 4 others are used for dedicated connections to the server machines for both security and load sharing reasons. The last Ethernet port is used over a dedicated fibre link to the undergraduate computing lab comprised of 30 NeXT stations and 5 or so Sun servers of various types. This allows the Computing Science students access to their campus home directory from the Computing Science lab as well as our server machines.

  A public lab of 40 NeXT stations will be added to one of the other Ethernet ports (along with a router) in the near future.

  The Auspex fileserver exports only the undergraduate partitions to the Computing Science lab, and the same will be true of the NeXT lab. None of the Auspex partitions are exported to the campus backbone. The reason for this is the potential insecurity of the NFS protocol. If a machine where a user can become root has access to the Ethernet carrying the NFS data, then that user can, without much difficulty, read any open file whether or not he should have access. We avoid this problem by a combination of things: not exporting important file systems (i.e.; those of faculty, grad students, and staff) to undergraduate labs. We also export the file systems to our server machines on a second Ethernet port and Ethernet that runs between the Auspex and the various servers that are all in a physically secure machine room. This means that an attacker has to either break root on one of our UNIX hosts, or to tap one of the private Ethernets in order to tap the NFS data.
- As noted earlier, there are 6 Server machines split by function and each with a dedicated Ethernet port for NFS traffic. In addition,

there is an RS6000 model 530 that is used for a group of 20 or so medium scale researchers, and 3 Sun SparcStation2s. One Sparc2 does NIS, NetNews, syslog, and terminal server security support. Another Sparc2 supports the DNS, the X.500 server, and is the campus mailhub. Each of these two machines have three Ethernet ports providing a private (non campus backbone) path for NIS and DNS services to the various other machines (on their NFS Ethernet ports). There is a third Sparc2 that is used to support the Xerox Soleil product that allows the Xerox 4090 printer to print either PostScript or ASCII data from a TCP/IP network.

This collection of servers, and dedicated Ethernet ports along with the Auspex file server spread the I/O across enough paths to allow things to work.

- Magnetic tape support currently resides on a VAX running VMS. This is because we were not able to find any UNIX software that could read all the various kinds of IBM standard labeled tapes. Data coming in on tape is read in to the VAX and then either transferred to 8mm tape on the VAX or ftped to the Auspex file server. At present the users that used to run their large data sets directly from tape, either store subsets on the file server disk and copy off tape as needed (a time consuming job) or do without. We will be installing a HP 20 gigabyte optical juke box onto the network in an attempt to provide users an automated (if low performance) way of storing their large data sets.
- An unanticipated advantage of having the Auspex file server turns out to be system backup. We have dedicated two Auspex disk drives to system backup. When a full backup is to be done, the two spare drives are mirrored to the active file system. This process takes about 20 minutes to make a duplicate copy of the online file system. When the mirror is complete, the mirror volume (now identical to the online volume) is detached from the mirror, which gives us a current snapshot of the online file system without having to bring the system down to single user mode. At this point the detached mirror disk is fsck'ed, and then backed up (presently using dump, but we may switch to doing a dd of the volume) to a labeled 8mm tape using some shell scripts we wrote to verify that the tape in the tape drive is the tape it should be, by reading the label before doing the dump. Daily incrementals are done on the running file system, so it is possible that we may lose up to a week's worth of data in the worst case. A shell script tape directory program enforces a dump cycle of 14 daily incremental tapes, and 4 level 0 weekly tapes, and 4 level 0 monthly tapes allowing us to recover data from up to 4 months in the past. This directory presents the required tape label to the backup script to make sure the correct tapes are being used for a backup.
- On the Sun Sparc server machines (NIS Server and DNS server), SunSoft's Backup Co-Pilot product is used to give us consistent dumps of the NIS and DNS machines without having to shutdown.
- Tape library services have more or less fallen back on a manual system. One VMS product has been tried, but was found to be unsatisfactory. We are being saved at the moment because tape use has fallen off significantly because of the difficulty of using the present tape operation.
- BitNet. Since we have some users that still need BitNet, we chose to implement BitNet on a VAX running VMS, using the Jnet BitNet software and a package called MX from RPI to implement a BitNet to TCP/IP gateway. Bitnet file transfers have to be done to and from an account on the VAX.
- The public X.25 interface is also done on the VAX, using a DEC X.25 Router2000 gateway server and a software package called PSI which provides access control and accounting (both of which we need in this application!).
- Printing. The central Xerox 4090 printing service has been replaced by 6 HP3si laser printers, driven from a Novell server that accepts jobs from all hosts (our UNIX hosts, plus Macs, PCs and other people's UNIX hosts on the campus backbone). The advantage to the central Novell server is that jobs can be spooled from anywhere to the server, and then be released via a release terminal (an IBM pc) that is next to each of the 6 distributed servers. This means that you can print your job from the closest of the printers. In addition, there is a magnetic card reader attached to each of the printers that has 5 cents decremented for each page printed on the printer meaning that the printing is accounted for as well. These cards can be charged up at several vending machines around campus allowing you to print if you run out of money after business hours.

  As mentioned earlier there is also a SparcStation running Xerox's Soleil software, that allows large and/or multicopy PostScript or ASCII jobs to be printed on the Xerox 4090 printer in theory at the rated engine speed of 92 pages per minute (in practice there are still a few bugs and we don't get anywhere near that throughput).

An interesting sidelight, the Xerox 4090 in MTS days used to print in the range of 600,000 to 800,000 pages a month (combined MTS and administrative VMS printing) at the present time it is doing about 200,000 pages a month, a large portion of that from the VAX. This drop off has not all been due to UNIX and the HP3si printers (the administration systems are also printing less); we believe that many of the course handouts that used to be done on the 4090 under MTS are either not being done (i.e. the file is made available online for the students to print themselves) or possibly are being done by the print shop. At least some of that demand has done other things for the summer but will return to the 4090 for the fall semester. (This may cause some interesting times for us in the fall.)

- Accounting. luckily the accounting for anything other than hard dollar costs (in practice public X.25 connections and printing) was ruled no longer required nor useful by the task force, so we are fine.
- Security. Mainframe operating systems in general and MTS in particular tend to have less security problems than UNIX. Some of the reasons are: the operating systems are proprietary and there is not general access to the operating system source code. Security tends to be far more important to the mainframe customers, and therefore more of a marketing issue than has been typical in the UNIX market place. A security breach on a mainframe system providing service to a large corporation can have a severe monetary impact, either through fraud or the loss of trade secrets.
- User directory. This was replaced by an X.500 directory system based on the QUIPO software, with locally written code to allow a lookup from Macs (and in the future PCs?). This runs on one of the Sparc2 servers (the DNS machine), with 64 megs of memory, and contains some 25,000 names at present.
- E-mail. The E-mail system was replaced with several options. Part of the problem here is that in the MTS case, since mail was on the central machine, mail could be accessed from any of the communication channels (directly connected terminal, dialup line, public X.25 connection etc.). The challenge with UNIX was to maintain this diversity of possible connections, while allowing mail access to move out to individual machines if desired. This was achieved by having a central mail host that accepts all incoming mail to an address of the form user_name@sfu.ca, and then directing that to the users central UNIX ID. If the user chooses to get mail on his or her personal machine, then they set up a .forward file to that machine on their central UNIX account directing mail to their personal machine. For people that choose to read mail on the central UNIX hosts, we support the Elm E-mail package on all of the central hosts. For people who choose to read their mail from a Mac we support the POP protocol with UNIX pop server on the mail host and for the Eudora Mac POP client. We have made some modifications to Eudora for use over dialup and public X.25 connections that have been fed back to the author. PCs are somewhat less supported; there is work going on to improve PCpop (and now NewPop) to get POP support on the PC. These changes too are being fed back to the authors. For those PC's connected via Novell we run the Clarkson Charon gateway on a Novell server to provide mail access. There is also work going on on providing interfaces from the various platforms to the X.500 directory service to provide E-mail address lookup from within the various mail packages.
- Conferencing. The Department Of Education commissioned a professor in the Communications Department whose speciality is electronic conferencing to examine the available conferencing systems and recommend one. Parti from Participate, was the package chosen, and it was installed on the general login server for teaching and research. Some of the former Forum users considered Parti to be a step backwards in conferencing and instead chose to do their conferencing using NetNews (and complained loudly and long about the money spent on Parti, but not to the committees that made the decision!).
- The PDP11 based terminal interface was replaced by three Annex3 terminal servers from Xylogics. We chose these terminal servers because the interface was UNIX like (as opposed to VMS like as some of the others were), they are capable of requiring a password for access, and the source to the UNIX side authentication daemon is supplied so we could (and did) modify the authentication to access the NIS server to verify passwords. We considered the Xyplex terminal servers which do authentication using the Kerberos protocol, but since Kerberos had been ruled out as too complex to be done within the deadlines imposed, this option was not chosen.
- Education. There was of course a massive education problem, since neither the systems people nor the user community were experienced in UNIX. This was solved by having the people that produced the MTS documentation create UNIX documentation. This takes the form of a series (more than 30 at present)

of single page handouts called "HowTo"s. Each HowTo covers a specific topic or UNIX command in detail or covers how something that the user used to do on MTS is now done under UNIX. The PostScript code for the HowTos are stored online along with a program that will send a copy to one of the printers, and preprinted copies are kept in racks in the Computing Center to be picked up. At present we provide ascii and PostScript versions of the HowTos that can be obtained by anonymous FTP and via the campus wide information service (currently provided by gopher).

- User account creation. This was done initially via shell scripts, using the NIS maps as a pseudo database for deciding whether or not a UNIX id or a mail alias was already in use for another account. All accounts were created, along with their home directory and initial files on the file server and a randomly generated (or supposedly randomly generated!) password set. This took a lot of time to process, but the scripts could be (and were) created in a couple of days, and were modified on the fly as things broke. Minimal function (due to time limitations) was put in, accounts could be created, but deletion and changes were not, these functions were to be added later.

What We Did Right

In hindsight, these things went well:

- Wired the campus. We were lucky enough to have technical people with a vision of the future, and management that felt that computing was an important part of the university to approve and fund the installation of the campus networking infrastructure. What is more, the design that was made when the choices of technology were no where near as clear cut as they are today have withstood the test of time, the infrastructure that we have in place is (as it was designed to) capable of supporting the new technologies as they emerge today.
- Committed to Cabletron equipment as the network vendor of choice for the network hardware. As with the cable plant, the Cabletron hubs have grown to support in a modular fashion the new technologies as they emerge, protecting our investment in network hardware. Their equipment has been very reliable, and the support has been good.
- Issued a "visual RFP". Since we had little idea how to capacity plan a UNIX system, we instead provided the vendors with our requirements as number of logged on users and what we expected them to be doing, and required the vendors to quote a system that met those requirements. Since our site was seen (and in hindsight, correctly so!) as an important example (at least in Canada) of how the conversion to UNIX would go, the vendors had a lot of incentive to make sure what they quoted would do the job.
- Purchased an Auspex NS5000 NFS file server. As pointed out earlier, the Auspex people appeared to be the only vendor that we talked to that both understood the I/O issue, and had a working installed system. Several points worked in Auspex's favor, as we mentioned: their marketing presentations were first rate, technical questions could either be answered by the presenter, or forwarded to the technical staff and an answer was forthcoming. The most compelling reason, however, was not the Auspex marketing folks (who could be expected to praise the product to the skies!) but in fact the customer reference sites. Many if not most of them, had more than one Auspex, strongly suggesting that they were happy with the machine. At one site we contacted, the Auspex maintainer had to think for a while (when we asked if the machine crashed a lot), then said some thing to the effect of "well, the only two I can think of were both power failures, and we were one of the beta sites and have had one for a year or so". To say the least, we were impressed by the obvious reliability of the box if it required thought to figure out when the last crash had been. At this point we have been running our NS5000 for a little more than a year, and by and large our experience has been the same, there have been problems that caused crashes, but not a large number of them.

  At the point that we bought the NS5000, we had not considered backup particularly (and therefore the mirrored backup didn't enter into the purchase decision). At this point, if we had the decision to make again, the mirrored backup capability that the Auspex can provide (especially given the conclusions in Elizabeth Zwicky's paper on the backup torture test) would be one more point in favor of buying an Auspex again.
- Selecting Silicon Graphics machines as the research machines and the general login server has worked well for us. The local SGI office provides us with excellent support, and the machines have performed as advertised. We would note that the general login server that was specified to support 80 users has seen a peak load of 110 users, and at that point response was still very good. This of course depends on a rigorous enforcement of no CPU bound jobs running on the login server, even a

single CPU bound job will tend to slow response on the system down, and several make it close to unusable.

- Set a UNIX novice to doing the documentation. This turned out to be a very good move, the documentation that was created covers the points that the novice user needs to know (since it was being written as the person writing it learned UNIX!), and yet is full and complete (our UNIX consultant learned of a command option he wasn't aware of while reading one of the HowTos). The HowTo collection is available via anonymous ftp (see the last paragraph), all we ask is to be given credit if you use them.
- Used VMS for several areas that UNIX supports poorly or not at all. We are lucky enough to have a VAX for the administrative computing, and in fact had a spare one that was surplus to the administrative system's needs but not worth much on the open market. This machine provides us with BitNet support (using Jnet and MX software packages as noted earlier), controlled, accountable X.25 public dial access using DEC PSI software. This has become important, because our public X.25 bill has tripled over what is was on MTS, largely due to the fact that MTS was line mode oriented, and a single packet contained a full line of characters, and UNIX is character oriented and likes remote echo (and therefore often costs two data packets per character rather than one packet per line), and we get charged by the kilopacket. The accounting software is allowing us to evaluate who is using the service and charge the cost back to the departments involved.

  In addition, this machine supports a bibliographic package called BRS that is being used to replace a package called Spires that ran on MTS but is not available on UNIX (we understand there is now a UNIX version of BRS). Perhaps the major use of VMS is to support the conversion of tapes. We have around 10,000 IBM labeled and unlabeled tapes from the 15 years of MTS in a variety of different formats (some of them custom). As well various of our researchers and our research data library get statistical and other data from other sites and governments on "IBM" style tapes and wish to process the data on UNIX. We were not able to find a UNIX based package that could read the various formats of IBM labeled tapes, but we were able to find several programs that run under VMS that can (with greater or lesser difficulty) read these tapes. As well, our VAX already has both 3420 (or "round") and a pair of SCSI 3480 (or "square") tapes, and the SCSI bus meant that adding an 8mm and cartridge tapes that UNIX supports could be done.

  The bottom line here is don't get hung up on doing everything on UNIX, look around and see if there is a more appropriate platform for doing some of the things UNIX can't.
- Using DEC's VAX Cluster Console package to provide the operator interface to all the UNIX hosts. This package again runs under VMS in this case on a VAXStation 3100 color workstation, and uses a Xyplex LAT terminal server in one of the Cabletron hubs to make dedicated connections to the terminal server ports that connect to most of the UNIX systems' console ports. This allows a single screen in the operator area to monitor and control all of the systems, it logs to disk all console messages for all machines, and allows the systems administrators to access the console of any UNIX box remotely (i.e. over a dial up line from home). In addition, there is a process that scans the console data as it comes in and that can be programmed to recognize error messages and produce an alarm condition on the operator's console (typically set the icon for the affected machine red, but it could page a system administrator if desired).

Some things, though, did not go so well:

- Magnetic tape support would have to head this list. There are two issues on tape support: transfer of data from off site sources for use on UNIX, and affordable, fast support for the processing of data sets that are too large to fit on disk. Of these two, the VAX solution detailed above works for the data interchange case, but at a large cost in skilled manpower to identify the tape type and arrange for the correct program to read and transfer it. In the MTS case, most of the time MTS could read the labels and figure out for itself what the tape format was and provide a data stream to the user without computing center intervention. The processing of large data sets directly from tape (bypassing disk) has not yet been solved. The performance of tape units on UNIX is a small (and unusable) fraction of the performance of a mainframe tape I/O system. Even if the performance were tolerable, there is no support in UNIX for tape management (in terms of controlling access to possibly sensitive data on the tape by user id), or support for the operator interface to request that the operator mount a tape on a tape drive and dedicate it to a user (we are aware of the packages around that will do this). We are at present installing a 20 gigabyte HP optical juke box system to attempt to solve the large data set side of this problem, and several sites in the MTS community are considering a joint effort to write a UNIX version of IBM tape support (since we had to do so for MTS, we know how).

- We underestimated the requirement for disk on UNIX. There was about 10 gigabytes of user data (out of a total of some 16 gigabytes in total) on MTS, so we bought a 10 gigabyte Auspex file server. We are presently at 18 gigabytes on the file server and will probably be adding more space. It looks to us like doubling your mainframe disk capacity when going to UNIX is just about right for planning purposes. One of the other MTS sites that had made some movement towards UNIX warned us that this is what they had found, and it appears they are correct. It is not clear to us why this is so, and it may only apply to sites running MTS (since both sites that have seen the requirement run MTS). Certainly some of the increase (perhaps all of it) has been caused by data that was stored and operated on from tape moving to disk in the UNIX case. There was no time to identify the cause, since the solution was simple: buy more disk.
- We didn't have time to make a proper user account creation system. This showed up as part of the next point, security (or more correctly the lack of it). Given the time frame imposed, this was unavoidable. There was simply not time to create such a system and still meet the deadlines, and the shell script and NIS database method worked – the accounts got created. The problems are several: the shell scripts never made it into sufficient control to allow the deletion of unused accounts, so many inactive accounts are active on the system. The random password generator wasn't quite random enough. It generated some 15 or so identical passwords for different accounts. In hind sight (which is of course always 20/20!), we should have checked that the just generated password did not match any previously assigned one. The correct (or at more correct) answer to this is to have a database (Sybase, in our case) hold all of the data about an account, including the id and mail alias to allow checking for duplicates when a new account is created, and only generate an account and assign a password when the user requests use of the account and identifies him or herself. This avoids the problem that we currently face of having many accounts that were never used until they were broken into and stolen.
- We failed to address the lack of UNIX security. This was probably our major and most expensive error. As we have pointed out before, MTS is a reasonably secure system, in addition, since there are only 8 MTS sites (7 now that we are gone!), there weren't a lot of people that knew what to do should they break a password and get in. Neither of these issues are true on UNIX, and especially not if you are connected to the Internet. Our lone UNIX expert is very experienced at securing UNIX systems, and therefore our hosts are reasonably secured, and we have been tightening up host security. We were aware that NIS and NFS were not very secure, but given the deadlines, we had to accept that they were the only way this was going to get implemented. AFS would have been a more secure and possibly better approach to file services, but it is not supported on the SGI machines, and as we pointed out earlier, there is NFS expertise on campus but no AFS expertise, so NFS was the safe choice. We have installed a copy of npasswd to force the selection of better passwords (after an initial run of crack broke some 800 passwords), and hope to install shadow passwords or Kerberos on all machines, at which point we will revalidate all users of the system and not automatically create an account for each user, but rather have an entry in the database that can instantly create an account, home directory, mail alias etc. when the user requests one and presents id.

  Chasing crackers and recovering from attacks on the system have chewed up a lot of staff time that no one expected to have to spend. There is an additional problem at SFU: Since MTS was a secure system, many of the users were used to storing confidential data on it and assume the same is true of UNIX. This is, of course, not so, it seems every day some new way to break security on a UNIX host is found and published.
- We did not (or could not) budget for more staff to support UNIX. The same number (45) of people that supported the central MTS system are now attempting to support 16 UNIX hosts from 3 (soon to be 4) different vendors, as well as provide UNIX support to the campus. This is clearly not possible, and what has fallen by the wayside is user support, since the manpower that used to be devoted to doing this is now going to keeping the UNIX systems running and maintained, and trying to document it all. There was a request from one of the lower level committees for an additional 50 staff members that would be distributed out to the the departments to provide UNIX support at the departmental level (there are some 60 people in the central site at present). Clearly, we needed to budget for more manpower to support a distributed computing effort (as opposed to a central

mainframe). I would note that we have not distributed very far, all the server machines that we support are in fact physically in the same machine room where that mainframe used to be. If we had in fact moved them out to the various departments, then there would have been an even stronger requirement for more manpower.

UNIX "Features" Attempted To Drown Us

We discovered a number of things in UNIX that work fine when you are dealing with a single workstation with from tens to hundreds of users, but that cause large problems when you are dealing with 18,000 to 25,000 users.

- If sendmail can not resolve the address to a user id, it proceeds to walk the NIS password map looking at the GCOS field trying to find a match so it can deliver the mail. When there are 18,000 accounts, several things happen: often, more than one message arrives at once and there are several processes attempting to walk the map. Since there are 18,000 entries in the map, a single walk takes a long time. Even one map walk provides a very heavy load on both the NIS server and the Ethernet over which it is communicating, overloading both the network and the NIS server and bringing the whole system to a grinding halt (as no other machine can get NIS service). The solution to this is fairly easy: throw away the vendor supplied sendmail, and install either IDA sendmail or Berkeley sendmail with the map walking code commented out. We chose to use Berkeley because we already had a sendmail.cf, and we had some troubles getting IDA to work (and the systems were dead NOW!).
- NeXT machines like to look over the full password file every 15 minutes or so, which again when it is 18,000 to 25,000 names long causes a heavy network load. When you have a whole lab of them boot at once: disaster. Again an easy fix, disable caching by setting the interval to 0. Finding this before your network melts down is much nicer than finding this is the reason that the whole system has been dead for the last hour or so.
- Our current password file is approaching the 26,000 mark (partly because the shell scripts that add accounts have no corresponding removal of dead accounts yet!), and at this level, you can cause the ndbm program to attempt to create a sparse file that is greater than 2 gigabytes (which of course fails). The solution we used here was to reduce the amount of text in each password entry (primarily changing "*disabled" to "*" in disabled password entries), but it indicates a limit that sites with large numbers of users need to be aware of.
- Due to our "one account accesses all" policy, we have exceeded the 32767 mark with some of our UNIX uids (since we have a common uid space for all machines on campus that use our file server). Several of the UNIX versions we have can not deal with a uid greater than 32767, and this is another limitation that sites with large numbers of users need to be aware of. We believe that we have most of our problems with this worked around. We would note that the maximum uid is 65535, so this may be a problem for very large sites.

Work in Progress

We are still working on some things:

- Modify the RPI database driven account management and generation system to work at our site. This is in progress now and should be complete by the time this paper is presented.
- Merge in a database driven resource accounting program that is being used to charge for printing on the Xerox 4090 and the departmental laser printers in the Computing Science department. RPI is interested in implementing this at their site.
- Install a 20 gigabyte HP optical jukebox and management software for the storage of very large research data sets. This is on order, and should be completed by the time this paper is presented.
- Install a public lab of 40 NeXT stations, and resolve the security and support issues surrounding this given no additional manpower. This should also be complete before this paper is presented.
- Implement the OnLine Consulting (OLC) from Project Athena at MIT. This is up now in test, without any of the standard Athena services (Zephyr, Hesiod, or Kerberos). The Discuss conferencing system used to store completed conversation logs has been replaced by an interface that posts the done logs to NetNews, and these changes will be sent out on the OLC developers' list.
- Convert our network from a bridged network to a routed network. For the last several years the networks of the 3 BC universities have been connected together as a single, large bridged Ethernet (with the intercampus connections being via T1 links). Apple tells us that we are the largest Appletalk / Ethertalk network that they are aware of. All three universities are now moving toward a fully routed topology. We are partly being driven to this by security concerns and problems: A Mac set to the same IP address as the CA*net

router blocked Internet access to all three universities until it was found, and of course, a lab of 40 NeXT machines is going to need some heavy securing (with being routed to the backbone being the start of that!).

- Install a donation of 20 X terminals from DEC. These X terminals use a virtual memory scheme to a DEC controller both to keep network traffic down and to allow less memory to be used in each X terminal. This will be a pilot project that may or may not lead to more X terminals being deployed.

Future Work

This is a list of the projects that we would like to undertake in the future to make computing life on our campus better; budget and time constraints will, of course, modify this list.

- Install Kerberos for network security. This is expected to be a large and hard to maintain job (unless of course the vendors start to support Kerberos).
- Install the shadow password suite on all of our systems. This is made more difficult by the use of NIS, so we may reconsider the use of NIS (although recent changes from Sun have made the master more secure.)
- Install secure 10BaseT hubs in the various computing labs to prevent the lab machines (Macs, PCs, NeXTs etc) from sniffing data from the Ethernet. A secure hub is one that only sends the data to the source port and the destination port; all the other ports get the packet, but the data portion is replaced by either random garbage or blanks so the data is not visible.
- Join in the effort to write an IBM tape support package for UNIX with several of the other MTS sites.

Conclusions

We have learned many things:

- There can be computing life after the mainframe: a diminished, I/O poor life perhaps, but life nonetheless. A large majority of the users on our campus use computing only for E-mail, and for them life has gotten better in many cases. At least some of the people out there use Eudora on the Mac, and never (and don't want to) sign on to UNIX. We discover this when Crack finds their password and they don't know how to sign on to change it. Replacements for the standard statistical packages (often a UNIX version of the same package) have kept the statistical users happy, with the exception of those who used large datasets from tape in the past. In general researchers running programs in Fortran were able to convert over to UNIX (not without effort and grumbling, but they were able to do it). There were a handful of users that had applications that are very MTS specific, and in fact a handful of them are still running on MTS at one of the other sites as being more efficient than converting. This is a point to keep in mind if you are converting, line up another site that is still running whatever you are converting from to satisfy those few users who really do need whatever you are converting from but only for a little while longer (making the cost of conversion too high!).
- If the full cost of the conversion is taken into account, we don't believe that a distributed UNIX system is cheaper than a mainframe; in fact due to the manpower increase, we think it is more expensive. (The full cost of the conversion includes the support cost of those people who chose to "do it for themselves" and run their own UNIX workstations, thus shifting the cost from the computing center budget into a departmental budget – possibly by funding a "research assistant" who just happens to be a full time UNIX system administrator).
- The Director Of Academic Computing Services here keeps saying that we should be giving away CPU cycles since they are cheap, and getting cheaper, and charging for the people required to maintain the systems who are expensive and getting more expensive, an exact reversal of the original reason for mainframes (when the hardware was expensive and the people were, relatively speaking, cheap).
- It is probably true that the increase in what the users can do with the new systems in terms of being able to run or share code from and with other sites on the network, graphics, the ease of use of e-mail more than justify the cost incurred by the conversion.
- Your VAX system programmer is your friend. Many of those tasks that are hard to do on UNIX will fit nicely on his or her VAX under VMS. Stress much this will increase his or her job security, just make sure that you forget to mention how much extra work this is going to be.

The real conclusion here is don't assume that a task can only be done by the UNIX system, look at the other resources that you have, and implement a service on the most appropriate platform. In our case, many of the instructional tasks that were once done on the mainframe are now done on either Macs or PCs in either public labs run by the computing center, or private labs owned and supported by a department that has sufficient demand to warrant the support expense. Be prepared to look at cost sharing plans where the department buys the machines and provides the space and the computing center provides a part of a full time person that is shared with other departmental labs because there isn't enough work to justify a full time position.

- If you make use of IBM labeled tapes you are almost certainly going to have problems moving to a UNIX system. Support for labeled tapes and mainframe like tape access control does not seem to exist on UNIX.
- If your site is connected to the Internet, you are almost certainly going to have intruder problems unless you take the time to guard against it before you install. Installing a password checking program like npasswd or passwd+ to force users to set good passwords is a good start. Get and apply the latest Sun patches that allow you to restrict access to your NIS server to hosts you select. Consider installing the shadow password suite if possible, consider installing a firewall machine between you and the Internet. Be aware that as shipped most UNIX boxes are wide open security wise; hire an experienced UNIX system administrator and listen to his advise when he suggests shutting off most of the "r" commands and sending mail to a program via sendmail. Even if the system would be easier to use with these programs enabled, they are not secure. Initially we did only the last of these, and thought our system was reasonably secure. We learned differently several times, and have lost a lot of staff time and are still losing a lot of staff time tracking security problems and attempting to close holes.

Availability

This section describes how to acquire many of the things described in this paper.

The HowTo documents (in PostScript and ASCII for Gopher) are available for anonymous ftp from ftpserver.sfu.ca in /pub/docs. All we ask is that you give SFU credit if you use them. The contact for HowTo questions (or contributions!) is Margaret Sharon (margaret@sfu.ca). The originals are in FrameMaker on the Mac.

The code that does mirrored backup to labeled tapes on the Auspex file server is available (if a bit rough!), contact Peter Van Epp (vanepp@sfu.ca) if interested.

The lpr filters and accounting software for the Xerox Soleil product can be made available if anyone else is using Soleil, again e-mail Peter Van Epp (vanepp@sfu.ca).

The database driven account management system is also probably available with consultation from RPI, please send e-mail to Richard Chycoski (richard@sfu.ca).

The Novell solutions, both for distributed printing and for reserving machine time in assignment labs can probably be made available send e-mail to Lionel Tolan (lionel@sfu.ca)

Advice, (possibly worth what you pay for it!), if you are facing a conversion like this we can answer e-mailed questions and arrange a limited number of site visits.

References

Baines, Urquhart (editors), *You Really Can Get There From Here: SFU's Migration to UNIX from MTS*, Proceedings Of The Community Workshop 92 (paper available via ftp from ftpserver.sfu.ca in /pub/ucspapers/RPIPaper92.ps.Z).

Zwicky, *Torture-testing Backup And Archive Programs: Things You Ought To Know But Probably Would Rather Not*, Proceedings USENIX LISA V, pp. 181-185.

Kolstad, *A Next Step In Backup And Restore Technology*, Proceedings USENIX LISA V, pp. 73-79.

Shumway, *Issues in On-line Backup*, Proceedings USENIX LISA V, pp. 81-87.

Polk, Kolstad, *Engineering A Commercial Backup Program*, Proceedings USENIX LISA V, pp. 97-103.

Author Information

Peter Van Epp has been a staff member of Computing Services Operations at Simon Fraser University for 5 years. Prior to joining SFU he spent 6 years at an airline as a systems programmer on the mainframe based reservations system. The previous 10 years were spent in a variety of real time control companies using mini and micro computers. He can be reached electronically at (vanepp@sfu.ca or vanepp@SFUVAX.Bitnet).

Bill Baines has been a staff member of Computing Services Operations at Simon Fraser University for 5 years. Prior to joining SFU he spent several years supporting administrative and real-time computing at a large industrial site where he learned more about performance management and capacity planning issues than he ever wanted to know. He has a B.Sc. degree from the University of Waterloo and can be reached electronically at (Bill@sfu.ca, or bill@SFUVAX.Bitnet).