Construction and Application of an x86 Beowulf Cluster

Susan Ditmore
TJHSST Computer Systems Lab
2004-2005

Abstract

A Beowulf (Linux) cluster of fifteen to twenty x86 Pentium II computers with automatic load sharing will be constructed for convenient use within the Computer Systems Lab. Computer classes such as the supercomputing course will be able to utilize the cluster.

Description of Methods, Algorithms, Procedures, and Process

Software

Both the Systemimager and Openmosix systems were implemented.
Openmosix is a kernel patch that automatically transfers and distributes processing jobs across a configured cluster of computers. It utilizes the Openmosix file system, designed specifically for the file transferring involved in distributed computing.
Systemimager is a utility that allows one to copy the "image" or Linux distribution and all configuration and software from one computer, called the golden client, onto an unlimited number of other computers. It also allows for "upgrades" of all the computers, downloading only what has changed.
In addition, dhcp3-server was implemented to provide DHCP service to the nodes of the cluster.

Design Criteria

The cluster required very specific design, due to the materials with which it would be constructed and the requests of the system administrators.
The operating system of choice for the cluster was Debian Linux, to ensure the greatest interoperability with the workstation computers and to ensure that current lab users would be able to transition easily to using th cluster.
Rather than pulling the Debian kernel, the vanilla kernel was downloaded and compiled with the OpenMosix patch applied. The afs files system patch was originally intended to be used in conjunction with the OpenMosix patch, but it was very difficult to compile in. In addition, OpenMosix uses its own filesystem optimized for load sharing.
When a job is started on a node computer, the node computer gradually migrates the job to other nodes. In order for it to migrate the other jobs and "get help" it must share the files necessary for completing the job with the other nodes. This is actually a choke point in terms of speed, and the Openmosix file system works to make this faster if it is installed on all the machines.
In the interest of space, the Debian image was left without x-windows system, because the harddrives of some of the computers were only 4 gigabytes.
The Systemimager system was to be utilized, its functionality explained above. In an effort to have a central server that hosts all possible images used in the lab, the server king was configured to host the cluster images in addition to the normal workstation computer images. The golden client selected was simply the first MCW workstation installed, hostname being akosmia.

A router box was constructed out of what was originally to be the imager server. It now serves as the head node to the cluster, hostnamed suntripsis. It serves as a dhcp3 server for matching mac addresses and hostnames; it defers DNS to macaroni. It is necessary to have a dhcp3 server for Systemimager, because when machines boot from floppy, they need to have their IP address automatically assigned via their mac address so they can communicate with the image server.

The router box, suntripsis, also serves as the head node of the cluster. Non administrators are not given accounts on any other machine; rather they are restricted access to this one node, which migrates jobs onto the others. Suntripsis is equipped with a large hard drive, so that users can store much information.

Interestingly enough, after the router box was configured, Openmosix did not operate on the small test cluster that was constructed. Openmosix reads the location of the other nodes from the /etc/openmosix.map file, but it often "rejects" this file as invalid for no valid reason. As a result, Openmosix falls back on an autodiscovery daemon named omdiscd, which broadcasts UDP messages across the network, "crying out" for other nodes. Since Openmosix would variably reject and accept our map file, and since the map file must be consistently read across the cluster in order for the nodes to see each other, it was decided to use the autodiscovery daemon at all times.

However, this also presented a difficulty, because the Openmosix autodiscovery daemon will always use the default route, so that only the public interfaces on the router and golden client which comprised the test cluster would be a part of the cluster. This was exactly the opposite of what was needed, because the cluster was only supposed to operate on the PRIVATE nodes.

The solution was to put the Openmosix cluster on a virtual LAN, routed through the server king. After a preliminary test, the autodiscovery daemon required some time to route through the network and find the test nodes, but the test cluster became operable.

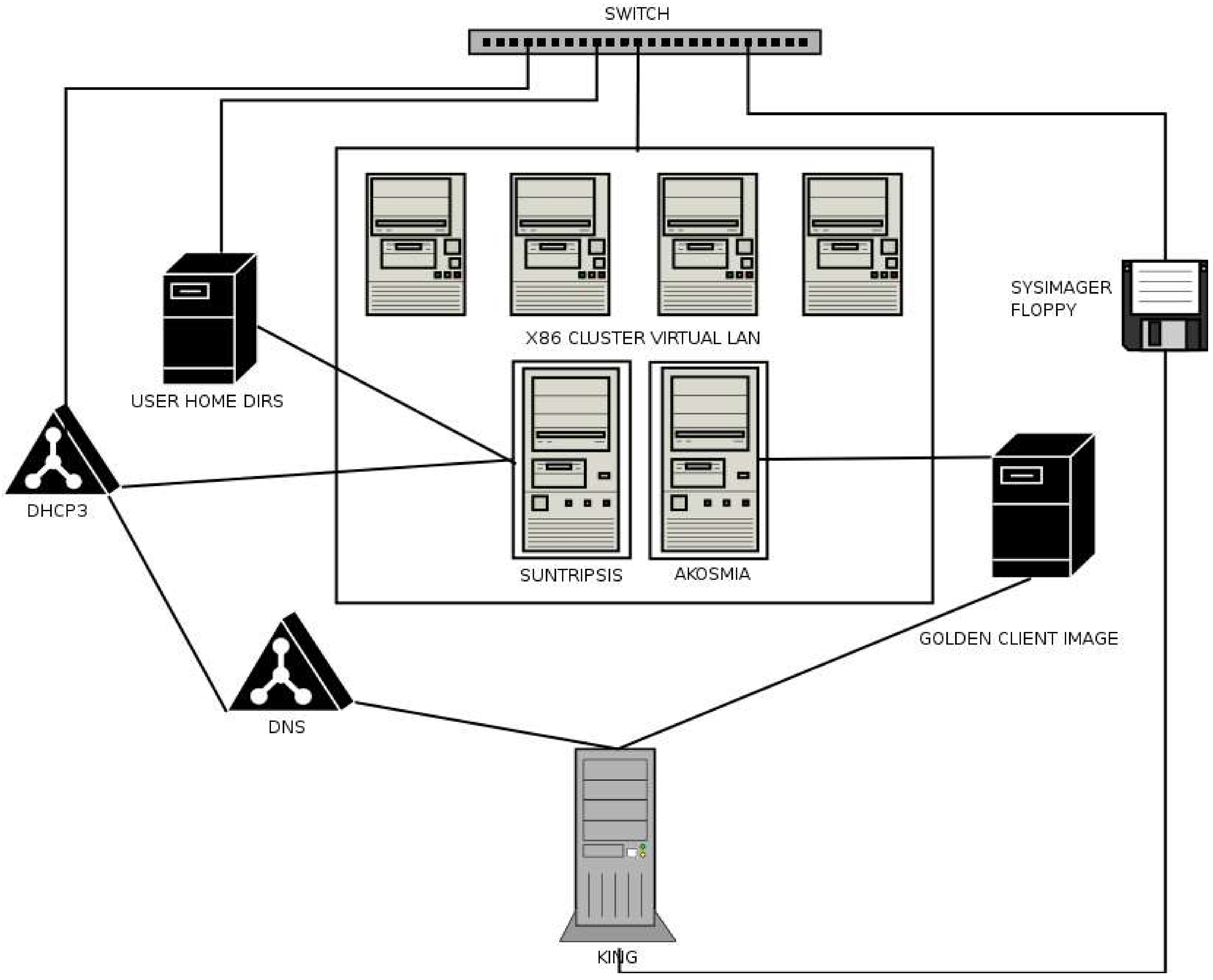
Procedures

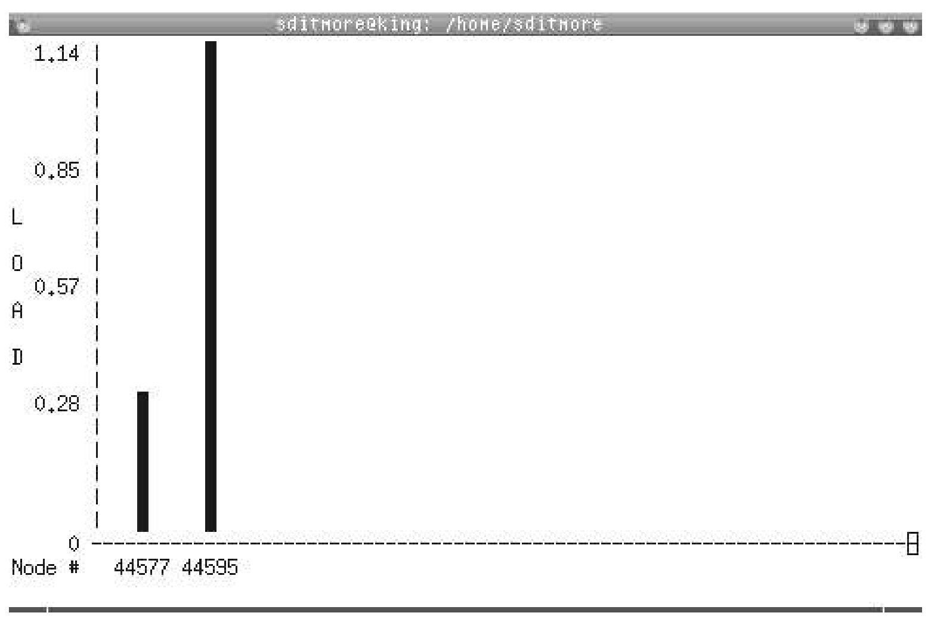
The first step of this project was to repair and clean the old Pentium IIs. This task actually required a fair amount of time, each computer requiring about half an hour to be cleaned and tested. They had so much dust inside them it would be fair to venture the guess that they had not been cleaned since their construction. About five of the machines were so broken that they could not be repaired; the remaining 16 survived and were added to the cluster.
After stacking the machines in the corner, it was necessary to find a place to store them. This was much harder than anticipated because there was a lot of controversy between various people concerning whether or not the cluster should be allowed in the machine room. It was, eventually, and a solid steel shelf was selected to store the machines.
Next the first level of the software for the cluster was constructed. Ie, the operating system. As explained above, a Debian kernel with the OpenMosix patch was constructed after only 23 tries!
The distribution was tested and configured on the golden client machine, akosmia.
Following this, the router had to be set up, to the specifications named above. Interestingly enough, the kernel did not have IP masquerading enabled, and so had to be rebuilt. It also had to be recompiled again to support the network interface cards inside the router.
Next the DHCP3 server was configured. The most difficult part of this was obtaining the mac addresses from machines with no viable operating system (ie, a very broken Windows 98). Each had to be separately booted on a debian boot disk and their mac address obtained using ifconfig. Several machines were discovered to be broken during this process, with their IDE motherboard controllers or AGP slots dying. Many also had bad cd drives, and so these had to be replaced.
At this point the virtual LAN as detailed above was constructed, as it was discovered the Openmosix daemon didn't work with routers very well.

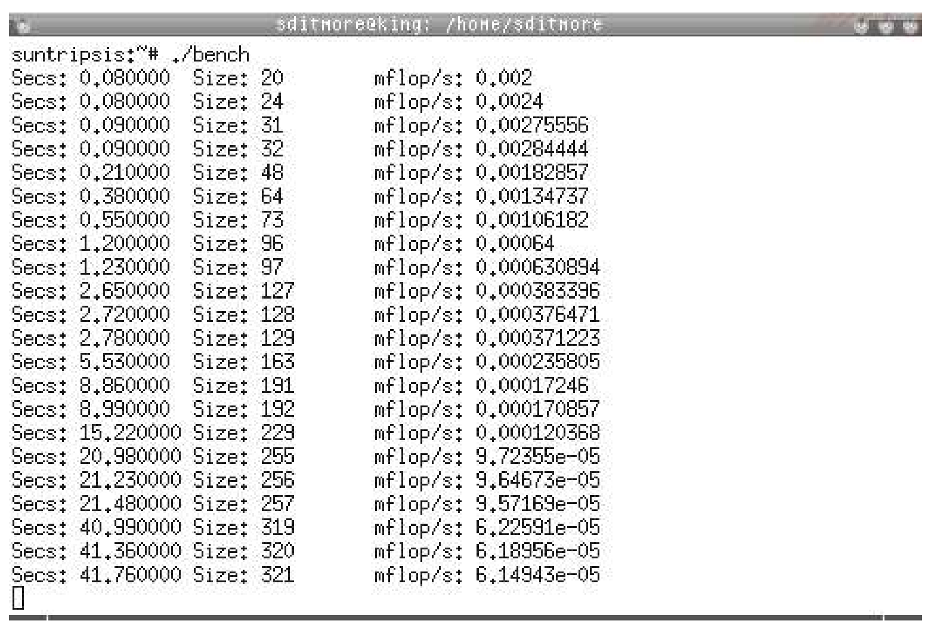
Next the image server was completed (this was the job of the system administrators), then the golden client was configured. \\
After that, system imager disks will be made and the distribution released to all nodes.\\
After that, various people will testdrive the cluster, and the image will be revised as needed. At this stage the cluster will be complete.

Background

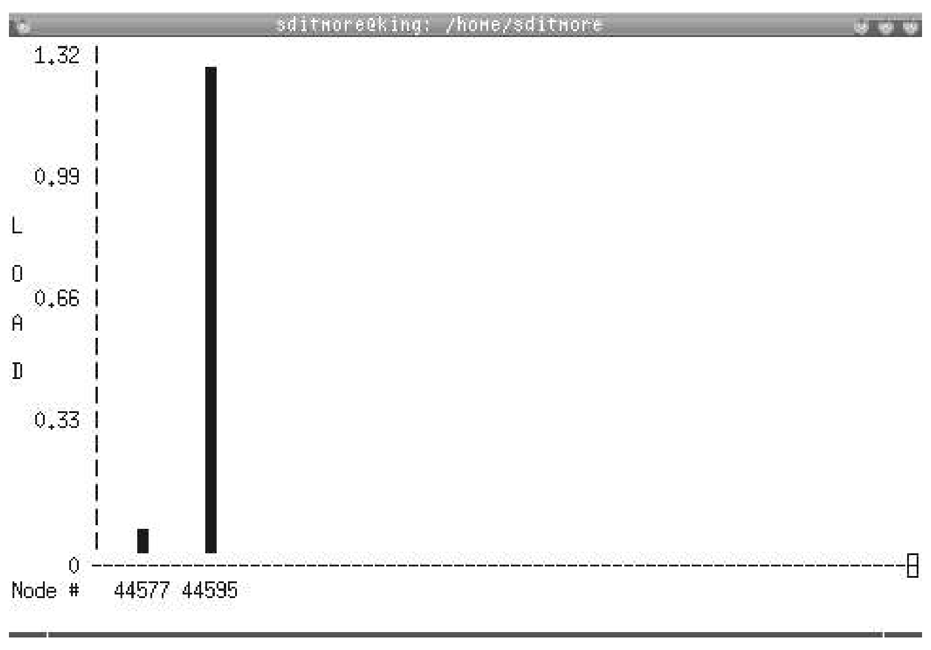
At the end of the summer of 2004 the Systems Lab was left with twenty or more Pentium II computers not powerful enough to be used as workstations. There was also a large supply of spare computer parts and spare almost working computers in the network supply closet.
The Systems Lab possessed a cluster of MIPs architecture computers but could not successfully port a working load distribution service to the cluster, as the most common and reliable services are tailored to x86 architecture machines.
There had been previous attempts to construct an x86 cluster, but due to various reasons the attempts were unsuccessful.

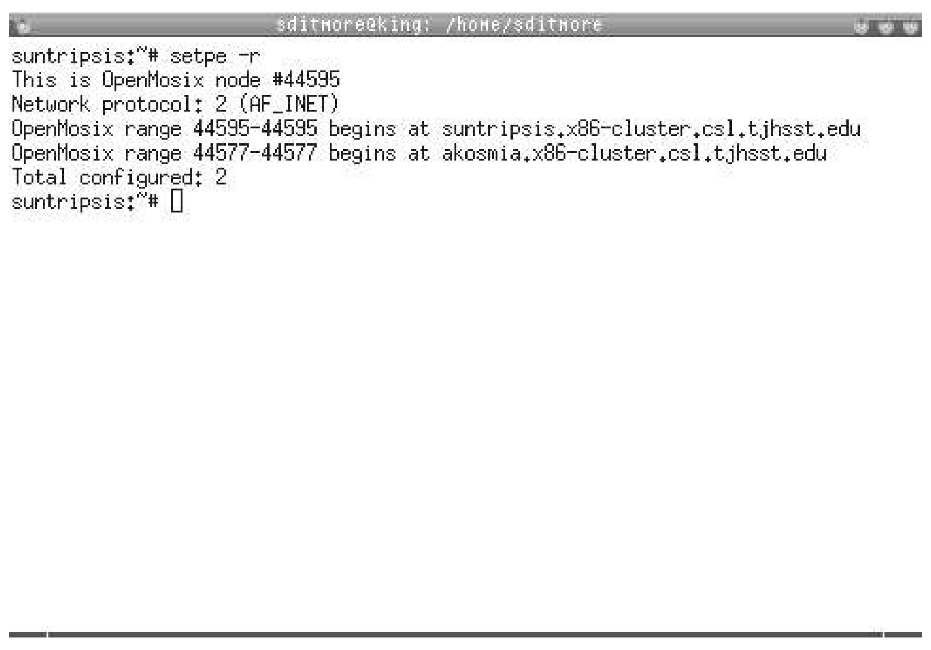






```
sditmore@king: /home/sditmore
suntripsis:~# ./bench
Secs: 0.080000  Size: 20        mflop/s: 0.002
Secs: 0.080000  Size: 24        mflop/s: 0.0024
Secs: 0.090000  Size: 31        mflop/s: 0.00275556
Secs: 0.090000  Size: 32        mflop/s: 0.00284444
Secs: 0.210000  Size: 48        mflop/s: 0.00182857
Secs: 0.380000  Size: 64        mflop/s: 0.00134737
Secs: 0.550000  Size: 73        mflop/s: 0.00106182
Secs: 1.200000  Size: 96        mflop/s: 0.00064
Secs: 1.230000  Size: 97        mflop/s: 0.000630894
Secs: 2.650000  Size: 127       mflop/s: 0.000383396
Secs: 2.720000  Size: 128       mflop/s: 0.000376471
Secs: 2.780000  Size: 129       mflop/s: 0.000371223
Secs: 5.530000  Size: 163       mflop/s: 0.000235805
Secs: 8.860000  Size: 191       mflop/s: 0.00017246
Secs: 8.990000  Size: 192       mflop/s: 0.000170857
Secs: 15.220000 Size: 229       mflop/s: 0.000120368
Secs: 20.980000 Size: 255       mflop/s: 9.72355e-05
Secs: 21.230000 Size: 256       mflop/s: 9.64673e-05
Secs: 21.480000 Size: 257       mflop/s: 9.57169e-05
Secs: 40.990000 Size: 319       mflop/s: 6.22591e-05
Secs: 41.360000 Size: 320       mflop/s: 6.18956e-05
Secs: 41.760000 Size: 321       mflop/s: 6.14943e-05
```





```
sditmore@king: /home/sditmore
suntripsis:~# setpe -r
This is OpenMosix node #44595
Network protocol: 2 (AF_INET)
OpenMosix range 44595-44595 begins at suntripsis.x86-cluster.csl.tjhsst.edu
OpenMosix range 44577-44577 begins at akosmia.x86-cluster.csl.tjhsst.edu
Total configured: 2
suntripsis:~#
```

Description of Results and Conclusions

So far each step has required a longer time than anticipated. This is due mostly to trouble with using old machines.
Many of the machines are in an extreme state of abuse and disrepair and so each one requires much attention and time to be revived. In addition the Systemimager concept that was prepared was later discarded by the system administrators for another system.
In addition it took 4 tries to find a successful routing box, as the previous three all had various odd issues such as locking up after a few hours or not displaying properly.
Preliminary work with the test cluster yielded an interesting result. The two test nodes of the cluster were a dual 400mhz Celeron "transformer" and a 300mhz Pentium II. If any work was started anywhere on the cluster, Openmosix would automatically shift all the work to the dual Celeron because it was so much faster

References

http://sourceforge.openmosix.net/
http://www.systemimager.org/
http://www.debian.org/