NFS and SMB Data Sharing Within a Heterogeneous Environment: a real world study

Alan Epps
Tektronix
Dr. Glenn Bailey
Tektronix
Douglas Glatz
Tektronix

Abstract:

A common problem encountered in a heterogeneous computing environment which includes both Unix & PC/Windows hosts is sharing data between the different operating systems. Any approach must take into account the different methods of authenticating users, file permissions, and network protocols. With 2000 PC desktops & 800 Unix users employing a variety of NT & Unix servers, the Unix support group at the Color Printing and Imaging Division (CPID) of Tektronix needed a robust system that was inexpensive, easy to administer, simple and effective to use for both PC and Unix users. Administering separate installations of local NFS clients on the PCs had proven to be problematic, causing us to look at a centralized server-based solution that could provide native PC file sharing via the SMB protocol suite. The potential solutions we looked at were: TotalNet by Syntax (sold by Sun as SunPC), Samba v1.9.18p10, NetServices v.1 & v.2 by Auspex, Network Appliance F230 series servers, and Sun's SunLink Server software v1 & v1.1. We compared performance, ease of use by end users, ease of administration, cost, support, training, scalability, and ease of integration into our current environment. As of this writing the conclusion was to use Samba.

1. Introduction:

The current trend of adding NT to already existing Unix environments, which has been gaining steam for the last two to three years, shows little sign of abating. The marketing machine that Microsoft has turned toward the back room server environment has propelled this trend at ever increasing speeds, making significant inroads into the traditionally Unix areas of file services, print services, web services, and even proxy and routing services. Discussions of 'easy of use' and 'initial training time' have had an impact on the management chains throughout our industries, driving Total Cost of Ownership (TCO) decisions from the initial purchase and setup side of the equation without always looking at 'impact costs' following deployment of a particular package. Support issues surrounding a heterogeneous environment of PCs in a Unix environment are often one of the impact costs that is lost in TCO calculations.

Due to increasing system diversity one of the larger impacts on our support organization has been the difficulty of file sharing between the Unix and PC platforms transparently from a user perspective. There are a large number of file sharing solutions available, from local NFS client software installed on the PC to SMB/NFS gateway software installed on a central server, to servers specifically designed to share files between the differing platforms. Traditionally Tektronix has used a locally installed NFS client on each PC, but as the number of PCs grew, and the number of operating systems grew, the difficulty in maintaining current versions, patches, and configurations on all of the machines became unwieldy. As this condition grew worse we began to look at the centralized SMB/NFS gateway designs, both from a pure software solution as well as from a dedicated designed server system.

The industry literature is full of test results and specifications for products and systems all claiming to be better than their competitors. All of these results use the optimal conditions for the particular product under test, maximizing its strengths and minimizing it weaknesses. While the numbers generated by this type of testing are interesting as a optimal end-point exercise, they are rarely indicative of the real world performance one can expect. Due to the need to justify any expenditures we made to upper management, any numbers we presented to them needed to be reproducible once we had the proposed system in place. As a consequence our testing was done using production machines, hooked into production networks, during the normal working day. While this introduced a large number of potential impacts on the test over which we had little to no control, by running each test multiple times and averaging the resulting numbers we ended up with an aggregate that should be reproducible once the system is put into full production. The test environment breaks down into three discrete elements: the test driver, the network configuration, and the platform under test.

2. Test Environment:

2.1 Test Driver:

The test driver was a Pentium II @233 megahertz with 128megs of RAM and a 3Com 10/100 NIC, the controller was a Pentium @166megahertz with 128 megs of RAM & a 10mb NIC. Originally the testing software was a perl script, written in house, that copied ~750 megs of user-generated data to and from the server under test. This script was eventually followed by NetBench version 5.01, a ZiffDavis testing suite recognized throughout the industry as a test driver for PC benchmarking. To maintain test continuity we continued to run the perl scripted test as well as the NetBench, and both numbers are presented herein.

The test data used by the perl script was ~750 megabytes of user files, ranging in size from 3k to 137megabytes, consisting of 1537 files in 283 directories. The initial perl script just copied the files up from the test driver to the server, with no random writes, and no reads, other than those used to verify the writes. This was eventually replaced by NetBench v.5.01.

2.2 Network Configuration:

The testing network was made up of a Cisco 300 series 100mb FastHub, a Synoptic 10mb hub, a Cisco 5500 Switch with a 48 port 10mb card & a 24 port 10/100mb card. The box under test was always given it's own switched port at 100mb, while the test driver was connected to the 10mb hub, which was in turn connected to a 10mb port on the switch. The controller was connected to a separate 10mb hub, which in turn was connected to a 10mb switched port on the 5500. All the machines were on the same logical network & VLAN, thus minimizing impacts on the test derived from packets between networks.

Diagram 1

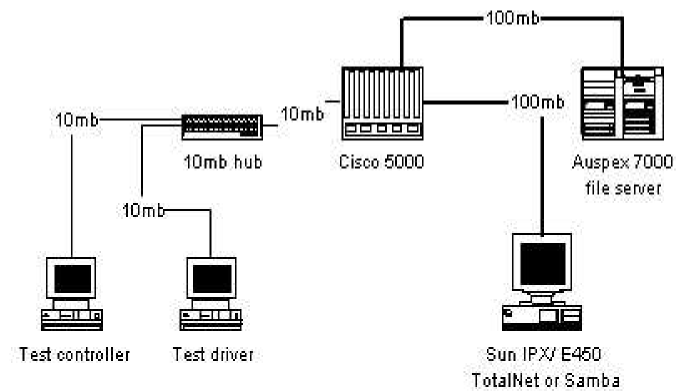
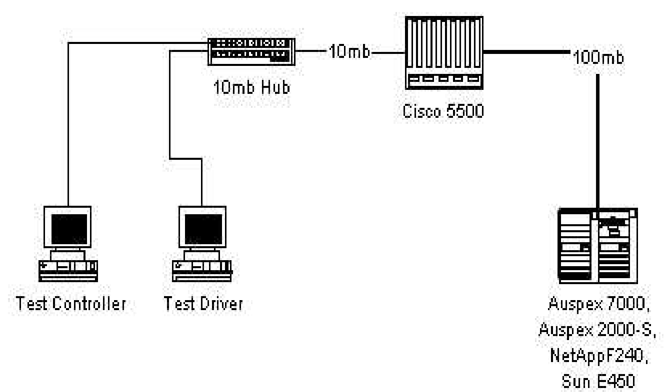


Diagram 2



2.3 Platforms Under Test:

<u>2.3.1 TotalNet Advanced Server by Syntax:</u>
Hardware configuration: Sun IPX, 128 MB RAM, quad 10/100mb Base-T NIC, Cycle5 processor upgrade
Software version: 5.2
See diagram 1

<u>2.3.2 Samba:</u>
Hardware configuration: Sun IPX, 128 MB RAM, quad 10/100mb Base-T NIC, Cycle5 processor upgrade
Software version: 1.9.18p10
See diagram 1

2.3.3 NetServices 1.0 beta by Auspex:
Hardware configuration: Auspex 7000/250, 512 MB RAM, 10/100mb Base-T NIC, 90Mhz HyperSPARC host processor
Software version: 1.0
See diagram 2

2.3.4 Network Appliance F230 server:
Hardware configuration: Network Appliance F230, 128 MB RAM, 4 MB NVRAM, 10/100mb Base-T NIC, Pentium II
Software version: ONTAP 5.1
See diagram 2

2.3.5 SunLink Server 1.0 beta by Sun:
Hardware configuration: Sun E450, 2 GB RAM, 4 x 300MHz CPUs, 10/100 Base-T NIC
Software version: 1.0
See diagram 2

2.3.6 SunLink Server 1.1 beta by Sun:
Hardware configuration: Sun E450, 2 GB RAM, 4 x 300MHz CPUs, 10/100 Base-T NIC
Software version: 1.1
See diagram 2

2.3.7 NetServices 2.0 beta by Auspex:
Hardware configuration: NS2000-P, 640 MB RAM,
Software version: 2.0
See diagram 2

2.3.8 Samba:
Hardware configuration: Sun E450, 2 GB RAM, 4 x 300MHz CPUs, 10/100 Base-T NIC
Software version: 1.9.18p10
See diagram 2

3. Test procedures:

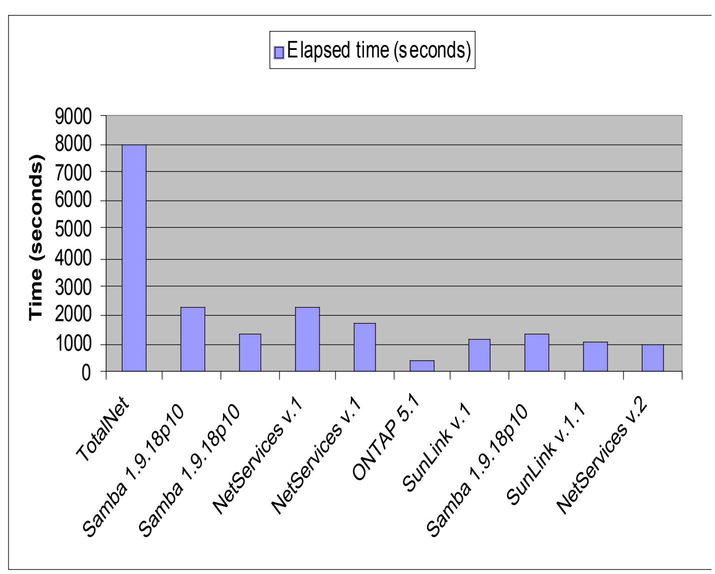
In all cases the host being tested was connected directly to a port on the 5500 switch, set to either 10mb or 100mb. A perl script was written that copied the test data to the server under test. Initially tests were run using a 10mb connection to the system under test, then unmounted and remounted using a 100mb connection when appropriate. Eventually we stopped doing the 10mb testing, under the assumption that the production system would be only connected to the switch via a 100mb port. Each test was run three times, each time during a different part of the workday, and the resulting time was averaged. In all cases the tests were run under working network load to reflect actual working conditions. The results are in elapsed seconds from start of write to completion of write.

4. Results:

The raw data results were:

Software tested	Elapsed time (seconds)
2.3.1 TotalNet 10mb	7980
2.3.2 Samba 1.9.18p10 10mb	2209
2.3.2 Samba 1.9.18p10 100mb	1385
2.3.3 NetServices v.1 10mb	2209
2.3.3 NetServices v.1 100mb	1694
2.3.4 ONTAP 5.1 100mb	426
2.3.5 SunLink v.1 100mb	1109
2.3.6 SunLink v.1.1 100mb	1078
2.3.7 NetServices v.2 100mb	910
2.3.8 Sambas 1.9.18p10 100mb	1291

A graph of the data is:



5. Sources of error

The potential sources of error introduced by our methods are many, ranging from individual server load variations to network broadcast storms or WINS elections. This is very deliberate, in that we wanted tests run in production networks on production servers while work was being conducted. Running each test multiple times throughout the work day should average out the larger impacts, yet still present a realistic spread of measured performance. By confining the network to our server room, and isolating most of the traffic within the backplane of the 5500 switch, a large amount of unrelated network traffic was avoided.

An additional source of error was the linearity of the original perl test script, which drastically favored the design of the Network Appliance architecture over any of the others in terms of raw performance numbers. An attempt was made to address this near the end of this project by using NetBench, but since all of the platforms had not been subjected to its test battery, the NetBench numbers are presented as additional information relative to the boxes tested with it.

6. Additional factors in the decision:

6.1 End user ease of use

For all of the solutions tested, end user impact was minimal. With no local client to install, configure, and maintain, all of our users have enjoyed an increase in stability, usability, and uptime. Additionally, the user community now only has one utility for accessing distributed resources rather than two or three, making their communication process with support personnel more clear and direct. Finally, the support staff has a much easier time tracing problems using only one protocol stack on the client PCs, rather than the two or three stacks that were previously installed.

6.2 Ease of administration

By centralizing our PC support structure onto a couple of servers we have largely eliminated the ongoing support costs relating to man hours spent visiting individual PCs to address configuration or maintenance issues. We have vastly simplified the troubleshooting process when problems do occur with the removal of the local NFS client.

6.3 Cost

As tested most of the configurations we explored ranged in price from $100,000.00 to $140,000.00 on the hardware solution side, including their related software cost. From a software solution side the prices ranged from free for the Samba & SunLink packages, to ~$12,4000.00 for the TotalNet package.

6.4 Support

At Tektronix we have an internal hardware support staff that is certified by our chosen vendors to do onsite maintenance and repair, with an average response time within a 2 hour time frame of problem notification. Computer Service and Support, or CSS, keeps a significant selection of spare parts and equipment on hand, but only for certain hardware configurations. One of the issues we had to take into account when evaluating hardware platforms was whether or not CSS was going to have a supply of spares on hand. In the case of both the Sun and Auspex hardware the spares were available.

6.5 Training

Our training costs should be reduced with the elimination of multiple client configurations and multiple client versions in concurrent use. Additionally, the training cost for troubleshooting should be reduced due to the simplification of client-server connections.

6.6 Scalability

In all cases the systems we evaluated should scale from the .5 terabytes currently in productions to the 1.5 - 2 terabyte data size we foresee in the next 18 - 24 months.

6.7 Ease of integration into our current environment

Given that CESA's job is primarily Unix server design and administration, the centralizing of PC support onto some sort of Unix server platform made the most sense in terms of ease of integration and training costs. This is mainly due to two factors: 1)the ability to leverage the experience base of the current administration staff and 2)the ability to leverage existing hardware for the initial phases of testing, and potentially early implementation as well.

7. Further avenues of work

We are continuing the testing process with an eye toward long term purchases for our server environment. A reality of our environment is that the data needs will exceed our current availability within the next six months, and a short term disk purchase will not address the long term scalability to the terabyte plus size we foresee in the next year or two. One of the primary goals is to re-write the current perl script to incorporate randomness into its write pattern, as well as random read structures. This change should then better reflect the across the board performance of all the platforms being tested, without favoring any particular design. Additionally, it will be written to run on multiple clients simultaneously, thus eliminating any discrepancies caused by some local load on the testing PCs.

8. Conclusion

All of the systems tested during this process would meet our needs as initially described, that of serving files between the PC and Unix environment without the need of a local NFS client on the PCs. One of the platforms tested showed a distinct advantage from a pure performance standpoint, but the largely proprietary nature of its architecture, as well as internal supportability issues relating to the hardware, kept it from being our choice. At the time of this writing the two choices we are going to pursue further are 1) Samba, running on existent hardware or on newer, larger hardware, or 2) Auspex NS2000-R. The rational for these choices includes familiarity with the hardware and software of the Auspex boxes, internal supportability by CSS of the Auspex or the Sun hardware, open source and support response time for the Samba package, and ease of implementation of either system.