Network Storage Solutions for Computer Clusters

Florin Bogdan MANOLACHE
Carnegie Mellon University
Email: florin@cmu.edu

Octavian RUSU
Alexandru Ioan Cuza University
Email: octavian@roedu.net

Adrian DUMITRASC
Carnegie Mellon University
Email: adriand@andrew.cmu.edu

Abstract

The network filesystem is one of the major performance bottlenecks for parallel computer clusters. This paper presents comparative results of different options, including filesystems like NFS and CIFS, hardware like dedicated fileserver and NAS, network parameters like jumboframes and connection protocols. Real world performance was simulated by transferring a mix of files of different sizes. The results show that dedicated fileservers over NFS offer by far the best performance and scalability without being CPU intensive. However, small variations in software parameters may lead to unexpectedly high performance penalty.

1 Introduction

Computational performance of computer clusters is ultimately limited by two factors, message passing efficiency and filesystem access speed. These factors can be alleviated by using proprietary hardware that can double the price of the cluster and have a high software maintenance cost. Alternatively, an initial optimization of the network setup and of the filesystem parameters can provide decent results.

This paper addresses the network filesystem access optimization. It studies several typical scenarios that can be implemented using standard components without increasing the startup cost of the cluster. The purpose is to find out which software configuration offers the best speed access to a remote filesystem hosted on a RAID array.

There are three typical major scenarios to be considered:

a. Dedicated fileserver: one of the cluster nodes has a RAID array attached, and is used exclusively for NFS sharing and account management, no real computation runs on it.

b. Headnode: this is the typical configuration used by the vendors to configure computer clusters; the headnode is used as fileserver, as job scheduler, and as working node as well.

c. NAS: the cluster nodes share a filesystem residing on a specialized appliance named Network Attached Storage (NAS); the NAS appliance manages a RAID and exports the filesystem via NFS or samba.

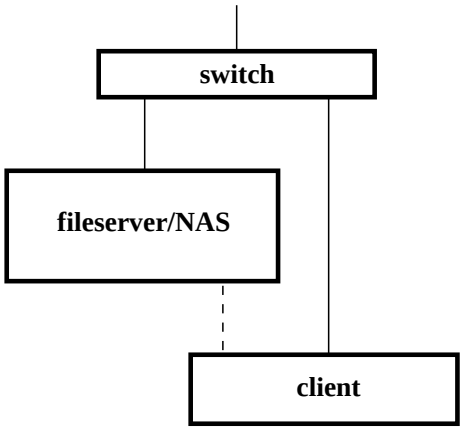

Figure 1: Experimental setup.

Other possible scenarios that were not explored so far involve distributed filesystems as well as filesystems optimized for parallel access [1].

The next section offers the details of the particular setup used for experimentation, as well as the procedure to generate the files that emulate live cluster conditions.

2 Experimental Setup

The setup of the experiment consists into a fileserver or a NAS connected to a client through a switch, as shown in Fig. 1. An alternative configuration idea was to eliminate the switch and use a direct connection, represented by the dotted line in Fig. 1. Since our experiment was designed to optimize clusters of computers, the direct connection was used just to test the delay introduced by the switch, and not as a mainstream alternative.

The components used for the measurements are typical parts provided by cluster vendors, in a standard configuration and having an average price.

The client is a computer in the following configuration: Intel Core 2 Quad CPU Q9300 at 2.5GHz, 2GB RAM, Gigabyte P35-DS3L motherboard, Realtek RTL8111/81688 PCI express gigabit network controller, SATA disk Seagate ST3320620SV 320GB, openSuSE 10.3 operating system.

The fileserver has the following components: Intel

Xeon X3230 CPU at 2.66GHz, 4GB RAM, Supermicro PDSMi motherboard, Intel Corporation 82573L gigabit Ethernet controller, 3ware Inc 9550SX SATA-RAID, 5TB RAID5 array, openSuSE 10.3 operating system.

The NAS is Netgear ReadyNAS NV+ RND4000 [2] containing four Seagate ST3750640NS disks in a RAID5 configuration. This NAS is able to export filesystems through CIFS or NFS. Tested options include jumboframes and sync for NFS.

The main switch used for tests was a Netgear gigabit smart switch GS724TS, software V1.0.1.4, jumbo frames enabled. For comparison purposes, several measurements were made using a direct connection, a Netgear ProSafe gigabit smart switch GS724Tv2, and a Linksys ESX588W unmanaged 10/100 switch. There was no performance difference between the transfers made on the two Netgear gigabit switches.

Files to be transferred have to be large enough to avoid different levels of cache mechanisms, and have a distribution that reflects the real activity of a cluster. Using a method described in [3], five groups of files were created. Each group had the total size of 4GB, and included files of the same size containing random data. The size and count of the files were 4x1GB, 16x256MB, 64x64MB, 256x16MB, 1024x4MB. To counteract the cache effects, a fifth 1GB file was transferred in the beginning of every test, but the timing results for that file were not considered for computing the averages.

Speed measurements can be done by using "time" applied to the "cp" command, or by using the throughput reported by "dd". While the two methods report different speeds, they are proportional to each other and consistent with the file sizes. The measurements presented in this paper were using the "time" utility.

Each test consisted into copying the set of files described above from the client to the fileserver (write test) and back (read test). Each such transfer generated a mean speed and a standard deviation for each file size. The average of the mean speeds for all file sizes was considered as the transfer speed for the given conditions.

Three types of remote filesystems were tried in the experiments: CIFS, NFS, and sshfs. While NFS and CIFS were stable and generated consistent results, our setup was unable to produce reliable measurements using sshfs. Frequent crashes made this filesystem unsuitable for production use. While the idea behind sshfs is very interesting, the implementation is new and not stable enough in openSuSE 10.3.

The transfer speed measurements showed that NFS is much more appropriate than CIFS for computer clusters. NFS development is very dynamic, and little information can be found about performance optimization of newer versions. Starting from "classic" textbooks [4], the experiments presented in this paper attempted to show what is best NFS configuration in current linux kernels, and how is the performance affected by different parameters.

The next section presents the results of the measurements for different network parameters and network filesystem scenarios.

3 Results

Data transfer speed is subject to fluctuations from a large array of factors. The most important factors we could identify were connected to cache mechanisms and to processes generating a high level of I/O or DMA requests. The experimental setup was trying to keep the fluctuations to a minimum by not running any other computation on the test system, and at the same time to keep the setup realistic by running most of the administrative processes that can be found on the typical cluster node. For that reason, it is considered normal to have a high standard deviation for the transfer speed of the small files. When the transfer speed of the large files exhibited an important spread, the experiments were repeated or checked for eventual problems. The standard deviation listed in this paper was calculated for the transfer of 1GB files.

Each data transfer iteration usually takes from 30 minutes up to a day. Only the results of successful transfers are presented. In the case of an unsuccessful transfer, the same conditions were reproduced on other systems to make sure the failure was not because of accidental conditions (power spike, bad block on disk, stray process on one of the computers, etc).

Jumbo frames used an MTU of 7200. That value seems to be preferred by the setup of the NAS, and was kept as reference for all the experiments involving jumbo frames.

Three stable configurations will be explored in the next subsections: NAS connected through CIFS, NAS connected through NFS, and fileserver connected through NFS. For each of these configurations we present a table with the transfer speeds. The first line of the table corresponds to a connection using default parameters, and is used as reference.

3.1 CIFS on NAS

The transfer speeds between CIFS on the NAS and the client running samba are presented in Table 1. The default settings produce a write speed of about 7.6MB/s and a read speed of about 9.5MB/s. The

auto-negotiating mechanism on the NAS sets up normal frames as suggested by line 3. Setting up jumbo frames importantly reduces the write speed, and has a low impact over the read speed, as shown on line 2. A direct connection, bypassing the switch, does not improve the speed, as shown on line 4. All the measurements were done using gigabit duplex Ethernet connections.

3.2 NFS on NAS

The transfer speeds between NFS on the NAS and the client are presented in Table 2. The default NFS options negotiated with the NAS server are the following: vers=3, rsize=131072, wsize=131072, hard, proto=tcp. These settings produce a write speed of about 10.6MB/s and a read speed of about 33MB/s. Line 2 shows that using jumbo frames slightly increases the write speed, but slightly reduces the read speed. Lines 3-5 suggest that a direct connection is as fast as a connection through the switch, with normal or jumbo frames. On line 6, we can see a 27% write speed penalty for using the "sync" NFS option. As expected, the read speed is not affected. Lines 7-10 repeat the previous experiments replacing the gigabit duplex Ethernet connection by a 100Mbps link, which becomes obviously saturated. The performance using an old 10/100 ESX588W unmanaged switch seems to be the same as the one offered by the GS724TS for this particular case. The 100Mbps link seems to increase the "sync" penalty to about 45

3.3 NFS on dedicated fileserver

The NFS transfer speeds between the dedicated fileserver and the client are presented in Table 3. The default NFS options negotiated with the fileserver are the following: vers=3,rsize=524288,wsize=524288,hard,proto=tcp. These settings produce a write speed of about 41MB/s and a read speed of about 52MB/s, importantly higher than the NAS can offer. Line 2 shows an important decrease of the write speed when the "async" option is used. A high performance penalty appears when the NFS "rsize" and "wsize" parameters are decreased to 8196, as shown on line 3. Various older documents [5] recommend these values, but re-using them for NFS version 3 or 4 can ruin the performance of the newer fileservers.

A comparison of transfer speeds via tcp and udp can be made on lines 4 and 5. It looks like NFS servers built in recent kernels are heavily optimized for tcp.

Line 6 shows the decrease in performance caused by the "noac" parameter. This parameter may be required by various MPI applications that have multiple processes accessing simultaneously the same file on an NFS filesystem. This is usually a bad programming practice and should be avoided. The "noac" option should be used only when there is no way around it. Mounting the NFS filesystem with a combination of options like "noac,rsize=8192,wsize=8192" caused the transfer time of an 1GB file to increase from 18 seconds to 19 hours.

Lines 7-9 show what happens when jumbo frames are involved. The only case when we see an improvement is the write speed with the "sync" option.

In a cluster, an NFS server can run on a dedicated computer or on a headnode that is also doing some other jobs. To emulate the headnode case on the fileserver, a set of benchmarks that will saturate the CPUs were run on it simultaneously with the file transfer. Lines 10-12 in Table 3 show the transfer speed while saturating 2, 3, and 4 of the 4 cores of the computer. Interestingly enough, the results show a slight performance increase as less cores are idle, with a maximum at 3 saturated cores out of 4. While doing these tests, the processor load was monitored. Typical CPU load of the active nfsd process was between 2-8of the transfer.

Lines 13-15 in Table 3 display the transfer speeds when 2 or 3 transfers are run simultaneously. The results show a very precise scaling of the speed with the number of transfers.

Lines 16-19 show the expected slow down when the Ethernet link is set to 100Mbps duplex. Interestingly enough, the performance penalty imposed by "noac" and "rsize=8192,wsize=8192" are not so high for this case. Any attempt to use jumbo frames at 100Mbps resulted in an unreliable network connection.

3.4 Filesystem Overhead

While read speeds don't depend much on the file size in any of the measurements described in this paper, write speed behaves as described in Table 4. The most filesize-independent speed is offered by NFS mounted from the NAS. Next is NFS mounted from the fileserver with regular frames, the speed ratio between 1GB and 4MB files is about 1.6. Using jumboframes strongly decreases the transfer speed of the small files. In that case, the speed ratio between 1GB and 4MB files is around 10, the same like for the samba transfer. These results show one more time that NFS in the linux kernel is heavily optimized for the default parameters.

4 Conclusions

This study suggests that the best compromise for a network filesystem in a computer cluster is NFS. The measurements show that dedicated fileservers are

Table 1: Transfer speed to NAS using CIFS

No.	frame	switch	write (MB/s)	σ_w (MB/s)	read (MB/s)	σ_r (MB/s)
1		GS724TS	7.63	0.033	9.46	0.140
2	jumbo	GS724TS	3.00	0.0064	9.18	0.056
3	autoneg	GS724TS	7.28	0.11	9.53	0.079
4	jumbo	-	3.00	0.039	9.23	0.096

Table 2: Transfer speed to NAS using NFS

No.	options	connection	frame	switch	write (MB/s)	σ_w (MB/s)	read (MB/s)	σ_r (MB/s)
1	async	1G		GS724TS	10.60	0.24	33.38	0.33
2	async	1G	jumbo	GS724TS	11.98	0.31	32.80	1.01
3	async	1G		-	10.77	0.13	33.49	0.45
4	async	1G	jumbo	-	11.93	0.30	32.26	0.86
5	sync	1G	jumbo	-	7.36	0.080	33.63	0.56
6	sync	1G	jumbo	GS724TS	7.46	0.022	32.22	1.85
7	async	100k		ESX588W	8.52	0.063	11.12	0.0076
8	sync	100k		ESX588W	4.59	0.025	11.10	0.010
9	async	100k		GS724TS	8.47	0.094	11.12	0.0063
10	sync	100k		GS724TS	4.64	0.024	11.12	0.0035

Table 3: Transfer speed to fileserver using NFS

No.	options	connection	frame	write (MB/s)	σ_w (MB/s)	read (MB/s)	σ_r (MB/s)
1	async	1G		40.72	2.12	51.89	3.58
2	sync	1G		5.79	0.10	51.27	0.65
3	async,rwsize	1G		16.79	1.81	26.96	1.40
4	async,tcp	1G		40.24	5.02	51.82	1.45
5	async,udp	1G		29.33	1.10	23.30	0.16
6	async,noac	1G		28.12	1.96	49.01	0.90
7	async	1G	jumbo	32.70	3.68	47.51	2.93
8	async,noac	1G	jumbo	7.50	0.068	50.19	1.72
9	async,udp	1G	jumbo	28.97	1.22	23.21	0.16
10	async,l2	1G		44.53	2.31	53.46	2.90
11	async,l3	1G		45.44	1.43	54.00	2.09
12	async,l4	1G		44.88	1.51	53.16	2.88
13	2xasync,l4	1G		23.40	1.55	26.55	1.78
14	2xasync	1G		21.87	2.49	27.32	1.03
15	3xasync	1G		14.92	0.87	17.80	1.36
16	async	100k		10.04	0.039	10.92	0.0035
17	sync	100k		4.18	0.040	10.92	0.0087
18	async,rwsize	100k		9.57	0.063	10.82	0.015
19	async,noac	100k		9.36	0.040	10.47	0.029

Table 4: Average write speed for different filesizes.

	fileserver NFS		fileserver jumbo NFS		NAS NFS		NAS CIFS	
File Size (MB)	write (MB/s)	σ_w (MB/s)	write (MB/s)	σ_w (MB/s)	write (MB/s)	σ_w (MB/s)	write (MB/s)	σ_w (MB/s)
1024	56.95	2.12	58.66	3.68	10.89	0.24	11.50	0.03
256	49.16	3.34	49.54	4.06	11.00	0.50	10.81	0.07
64	31.28	3.29	31.62	2.62	10.53	0.78	9.42	1.29
16	31.64	7.00	17.22	7.95	10.49	0.63	5.01	4.07
4	34.57	10.12	6.45	3.19	10.10	0.92	1.41	0.82

faster than NAS appliances. The fastest configuration was in all cases the default one, which is to be expected. Altering mount parameters and other options lead to a quick decrease of the transfer speed, and should be done only for a good reason. The performance of the NFS server is not affected much by the CPU load, so using the fileserver as a headnode is a feasible scenario as well from the point of view of remote file access.

References

[1] Jeff Layton, "File Systems for HPC Clusters", *Linux magazine*, Vol. 9, Issue 11, pp. 30-32, 2007.

[2] Fernando Cassia, "Gigabit LinkStation is power user's dream NAS drive", The Inquirer, Mar. 29, 2006, http://www.theinquirer.net/en/inquirer/news/2006/03/29/gigabit-linkstation-is-power-users-dream-nas-drive

[3] Octavian Rusu, Paul Gasner, Florin B. Manolache, "Comparative Study of Wired and Wireless Networks Capability", *Fourth RoEduNet International Conference*, Tg. Mures, Romania, 2005.

[4] Dave Olker, *Optimizing NFS Performance: Tuning and Troubleshooting NFS on HP-UX Systems*, Prentice Hall, 2002.

[5] Tavis Barr, Nicolai Langfeldt, Seth Vidal *Linux NFS-HOWTO*, Chapt. 5, 2000, http://www.linuxdocs.org/HOWTOs/NFS-HOWTO