Preliminary results using IDE drives in a hardware RAID configuration

February 22, 2000

1 System setup

The system contains the following components:

- Tyan 1832DL (Tiger-100) Dual processor motherboard ($177)
- 2 boxed Pentium III 550 MHz Katami (0.25 micron) processors ($646)
- 2 128MB generic PC100 SDRAM chips ($258)
- 1 NetGear Gigabit network card ($308)
- 1 NetGear 10/100 network card ($22)
- 1 Diamond Viper V330 video card ($34)
- 1 3ware raid controller ($450)
- 8 40GB Maxtor DiamondMax disk drives ($2170)
- 1 13GB Maxtor DiamondMax disk drive ($130)
- 1 floppy disk drive ($18)
- 1 Antec 4U case (7" x 19" x 19") with 400W power supply, 2 front fans, and sliding rails ($431)
- 1 extra front fan ($5)

The total cost of the system including shipping was $4650 when it was purchased at the end of January, 2000. The components were assembled by physicists at Vanderbilt University. The assembled product is pictured in Figs. 1 and 2. Because there was not enough room for a permanent CDROM, RedHat Linux 6.1 was installed by attaching an ATAPI CDROM externally to the second IDE bus with the cover off. The RAID configuration was quickly accomplished using the 3ware BIOS. The necessary drivers are configured as modules which are inserted into the default kernel by a script which runs last in `rc5.d`. This script inserts the module (using `insmod`) and mounts the device (which is `/dev/sda1`). A standard `mke2fs` makes the file system and the server is ready to go. The 3ware RAID card uses IDE drives with its own protocol and switching network.

2 System performance

We performed a variety of performance measurements on this system. We first performed local reads and writes using only the server. We used the benchmark *iozone* as well as standard UNIX `cp` commands to determine total throughput. The default speed of reading and writing single files seems to be between 50-60 MB/s. Table 1 shows the results for copying single files to and from the RAID array. In the case of UNIX `cp` the transmitting and receiving was done using `/dev/zero` and `/dev/null`, respectively. It can be clearly seen that buffering allows files less than 128 MB to be written at approximately the PCI bus speed ($\sim$133 MB/s).

Figure 1: Top/front view of system. Four 40GB EIDE disk can be seen on the left with the 3ware card between the last two on the left. Four others are stacked horizontally behind the homemade fan/copper plate arrangement. The 13GB system disk is next to the floppy drive.

File size	Write speed (MB/s)		Read speed (MB/s)	
(MB)	UNIX cp	iozone	UNIX cp	iozone
4		60		60
16	120	60	45	60
64	120	60	45	60
128	100	55	50	55
256	60	50	50	50
512	40	40	50	50
1024	25	45	55	55

Table 1: Single copies (iozone copies are with 16kB records).

Figure 2: Top/side view of the system. Four 40GB EIDE disk can be seen on the right with the 3ware card between the bottom two. One of the CPUs can be seen on the top left above the four cards (1 AGP graphics card, 1 100Mb NIC, 1 1000Mb NIC, and the 3ware card).

The results for writing and reading multiple files are shown in Table 2. Here there is clear disagreement between the *iozone* benchmark and UNIX cp method. It is clear, however, that increasing the number of simultaneous operations severly diminishes total througput. Once 8 processes are simultaneously accessing the RAID array, the total throughput is down to less than 20 MB/s.

Finally, we test this machine as a server of disks to many other machines. In this setup we attach up to 8 "worker" nodes on the same switch as the server. The worker nodes have Fast Ethernet (100 Mb/s) network cards and UDMA disks with a maximum transfer rate of $\sim$12 MB/s. The server is connected to the switch via a Gigabit (1000 Mb/s) network card. The results are shown in Table 3 and are all obtained using UNIX commands (`rcp` and `cp`). These tests provide information about network limitations, local disk speed limitations, RAID array limitations and even CPU limitations. All commands are performed on the individual nodes and all writes are done from data in memory (avoiding the delay of local disk). In test 1, when only one or two nodes are attached it is clear the limitation is in the Fast Ethernet card. When three or more nodes are reading from the RAID array, however, it is the RAID array which is the limiting factor. In test 2, no disk should be involved in the operation. Therefore the maximum throughput of 37 MB/s must be due to some other limitation of the server or switch; most likely the Gigabit network card. In test 3 we see that `rcp` writes to the server disk peak at 25 MB/s, significantly below the 37 MB/s of test 3. Tests 5-7 show that NFS operations are approximately 50% slower than the equivalent `rcp` operation.

File size (MB)	Number of processes	Write speed (MB/s)		Read speed (MB/s)	
		UNIX cp	iozone	UNIX cp	iozone
16	1		60	50	60
16	2		90	40	90
16	3		90	40	100
16	4		90	40	100
16	5		90	30	90
16	6		90	20	80
16	7		80	20	70
16	8		80	20	80
64	1		60	45	60
64	2		90	40	80
64	3		70	40	40
64	4		70	40	40
64	5		50	45	30
64	6		50	45	20
64	7		40	45	15
64	8		30	45	15
128	1		55	50	55
128	2		65	40	20
128	3		45	40	20
128	4		35	40	20
128	5		30	35	20
128	6		25	25	15
128	7		25	20	15
128	8		25	20	15

Table 2: Multiple copies (iozone copies are with 16kB records). Transfer rates are total throughput.

	Number of nodes	1	2	3	4	5	6	7	8
Test	Operation	Total throughput (MB/s)							
1	RCP read from server disk to worker /dev/null	11	21	29	31	32	30	27	24
2	RCP write from worker to server /dev/null	11	22	32	35	37	37	37	37
3	RCP write from worker to server disk	11	21	24	24	25	25	25	25
4	NFS read from server disk to worker /dev/null	7.8	15	18	22	24	25	28	23
5	NFS write from worker to server disk	7.0	12	16	16	17	18	18	17

Table 3: Server performance. RCP and NFS copies of 128 MB files to and from the server). All writes are from cached files on worker nodes. No files are cached on the server. Transfer rates are total throughput.

Although these numbers look impressive, there are hints that this system does not scale as well as we might like. Preliminary indications are that running 8 jobs reading at full speed and writing 2% of the data (both with NFS) are enough to slow the total throughput to just 3 MB/s. More investigation is underway. In the future we hope to compare this to the results of a hardware SCSI solution.