Taken from Danesh A, Gautam D, eds. Special Edition Using Linux System Administration, Que Publishing, 2000.

(A) Chapter 10 - Ram Samudrala

(B) Installing and using RAID

(C) Understanding RAID

(D) Background

RAID is an acronym for Redundant Array of Inexpensive (or Independent) Disks. The term was first coined by Patterson, Gibson, and Katz to describe how a number of inexpensive/independent disks could be combined (see Figure 10.1) into what appears as a single storage unit to the computer with performance similar to large disks (Patterson DA, Gibson G, and Katz RH. ``A Case for Redundant Arrays of Inexpensive Disks (RAID)'' Technical Report UCB/CSD 87/391, University of California, Berkeley, CA 1987).

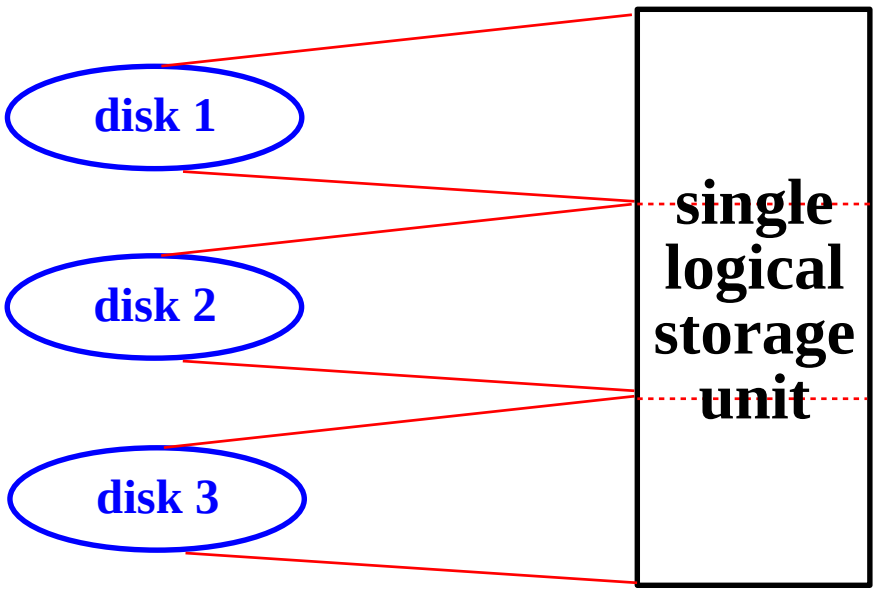

Figure 101: Illustration of a RAID system where a number of independent disks are combined to form a single logical storage unit (RAID).

(D) Why you should use RAID

There are several reasons to use RAID, the primary ones being increased storage capacity, fault tolerance during disk failure due to its redundancy, and speed. The economic costs of a disk failure

can be extremely high (not just because of down time, but also because of lost productivity and recovery costs). RAID systems generally enable one to recover from single disk failures quickly and transparently, without any down time. Thus RAID is a good option for those who want an additional support layer to shore up a traditional backup system and do not want to always bother with restoring backups every time a disk fails

(D) Striping

Fundamental to understanding how RAID works is the concept of ‘‘data striping’’. Data striping involves splitting up the storage space of each drive into small chunks of data (or stripes) which are then interleaved across the different drives in the array (see Figure 10.2). The combined data is composed of stripes in succession from each drive. The stripes can be as small as one sector (512 bytes) or as large as several megabytes, depending on environment the array is being used in.

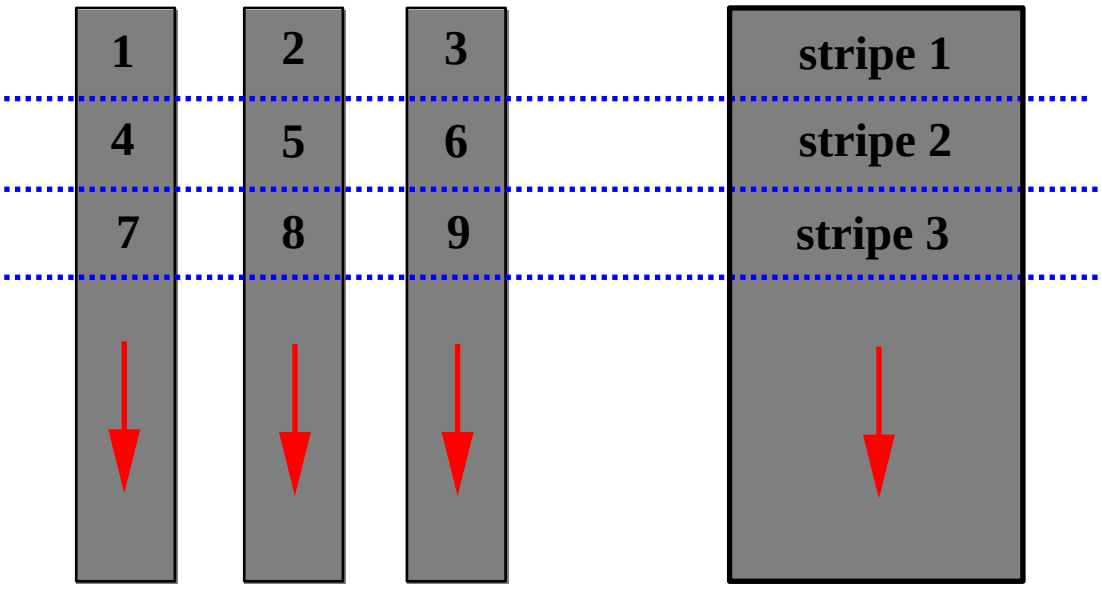

Figure 102: Data striping across several disk drives. Blocks of data from each disk are grouped together sequentially to create stripes of data in the array.

(D) Different RAID architectures

There are six recognised types of RAID architectures, RAID-0 through RAID-5. Each provides a different type of disk fault-tolerance and offers different trade-offs in features and performance.

Fault-tolerance is key to a RAID system because the average failure time of the array is equal to the average time between failures of each drive divided by the total number of drives. Therefore, it is essential there be some redundancy in the data storage to make up for the lower average time between failures for the array as a whole. The level of fault-tolerance also affects the performance and features of the array. Each type of RAID is described below, and Table 10.1 compares the different types of RAID and their features at a glance.

- RAID-0 arrays are sets of striped disk drives with no redundancy (thus it doesn't completely fit the RAID acronym) in the data storage, and consequently without any fault tolerance. These arrays are commonly configured with large stripes for applications requiring intensive I/O, since the splitting of data across drives results in higher throughput and data storage efficiency. However, a single drive failure will cause the entire array to crash. This level is also known as disk striping, and is the fastest and most efficient array type.
- RAID-1 arrays consists of pairs of disk drives that store duplicate data. Thus there is complete redundancy: if one drive fails, the other one of the pair is still available. Level 1 arrays tend to perform better on reads (since both drives can perform reads simultaneously) than on writes (since every write must be done on both drives of the pair). Striping is not used with a single pair of drives. This level offers decent performance and fault tolerance, but has the least storage efficiency of any RAID level. This level is also known as disk mirroring, and is most useful for performance-critical fault-tolerant environments.
- RAID-2 arrays are similar to RAID-0 arrays except that error correcting (ECC) information is stored to achieve disk fault tolerance. This level is seldom used today since modern disk drives have ECC information embedded within each sector.
- RAID-3 arrays are similar to RAID-2 arrays except that a single drive contains the parity information, and the data is striped at a byte level across several drives. As a consequence, RAID-3 can perform only one I/O operation at a time, limiting its use to single user systems. However, it is good for data-intensive single-user environments which access long sequential records.
- RAID-4 arrays are similar to RAID-3 arrays with the exception of large stripe sizes. The performance of this level is very good

for reading (on the same level as for RAID-0), and not so good for writes (since the parity information has to be updated each time). Large sequential writes are fairly fast, but slow random writes can be very slow. All the parity data is stored on a single drive and therefore this level is cost efficient.

- RAID-5 arrays circumvent the bottleneck caused in RAID-4 due to storing parity in one drive by evenly distributing the parity information among all the drives in the array (see Figure 10.3). Thus multiple write operations can be processed simultaneously, resulting in improved performance. The equivalent of one drive's storage space is used up to store the array's parity data. RAID-5 arrays are extremely flexible because they can be configured with either large-stripes (for multi-tasking and multi-user environments) or small-stripes (for maximising transfer rates). The cost efficiency of this level is the same as for RAID-4. This level is one of the most commonly used levels today, and is the best choice in multi-user environments that are not write-sensitive.

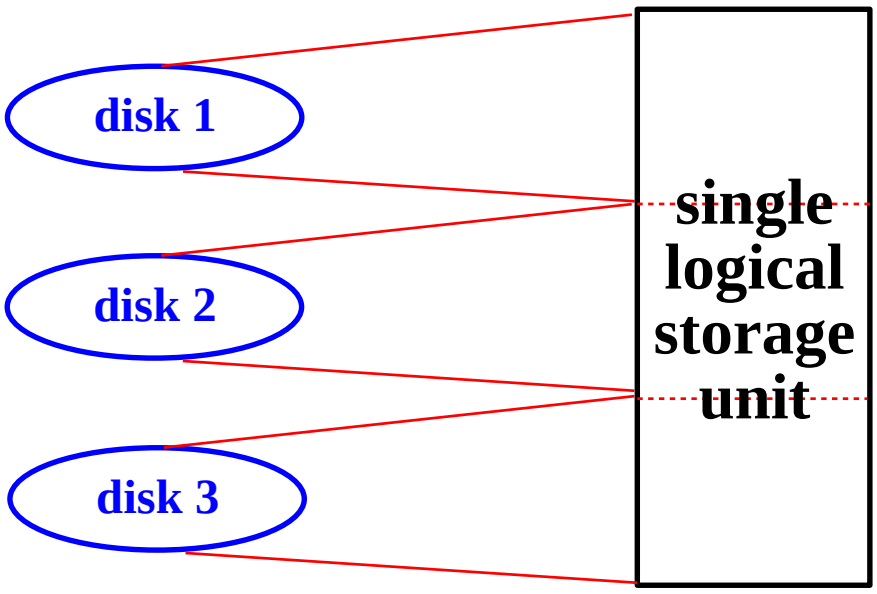

Figure 103: Illustration of a RAID-5 array, where the parity information is distributed across all the drives. Each write requires the parity information to be updated on all drives, but reads can occur simultaneously on every drive.

Table 10.1 [em] Comparison of features of the different RAID levels

Feature	RAID-0	RAID-1	RAID-2	RAID-3	RAID-4	RAID-5
Fault tolerance	None	Best	Good	Good	Good	Good
Storage efficiency	Best	Worst	Good	Good	Good	Good
Read performance	Best	Good	Good	Good	Good	Good
Write performance	Best	Good	OK	OK	OK	OK

For more information on RAID, refer to the What is RAID? FAQ by Michael Neuffer at http://www.uni-mainz.de/ neuffer/scsi/what_is_raid.html and the DPT Understanding RAID primer at http://www.dpt.com/library/library.html

(D) RAID implementations

RAID can be implemented in the hardware, in the form of specialised controllers that manage the array, or in software, as part of the kernel that is layered between the low-level disk driver and the filesystem above it.

(E) Hardware RAID

Hardware-based RAID can take on two forms: host-based controllers, where an adapter card plugs into an ISA/EISA/PCI slot in your computer, or SCSI-to-SCSI controllers where an external ‘‘box’’ that is connected to the host via a normal SCSI controller and appears to the host as a single disk.

In the former case, the adapter/controller card manages the RAID independently from the host. The primary advantage of using a controller card has to do with its ability to utilise multiple SCSI channels to read and write from the RAID array, thus increasing the transfer rate over the SCSI bus. In the latter case, all the RAID handling is accomplished by a machine containing both a controller and an external disk subsystem. A single SCSI channel must be used to control the RAID and this can create a bottleneck. Thus host-based controllers are the preferred choice for a hardware RAID system.

(E) Software RAID

RAID can also be implemented purely in software, without any requirement of extra hardware, by a set of kernel modules and management utilities. The MD driver under Linux is one example of such a RAID implementation (which implements levels 0, 1, 4, and 5).

The Linux RAID driver is implemented as a kernel layer between the block-device interface (upon which the filesystem resides) and the low-level disk drivers (for IDE and SCSI drivers). For more information, see the Software-RAID HOWTO at:
http://www.linuxdoc.org/HOWTO/mini/Software-RAID.html

(E) Hardware vs. Software RAID

Software-based arrays run like any other application on the system and therefore will consume a lot of CPU cycles. They are also operating system dependent. The performance of a software-based RAID is greatly affected by the other applications that are running concurrently on the system. In contrast, hardware-based arrays generally do not depend on the CPU performance and load of the host machine.

Hardware-based RAID schemes have little in common with software-based RAID systems except in terms of the array functionality.
Hardware-based arrays have true multi-tasking, since the processor in the controller can execute multiple instructions independent of the operating system type and function. Hardware-based RAID controllers also have an on-board memory cache that can allow high transfer rates into the large controller cache. Hardware-based RAID systems are more reliable since no special software is not required to boot the system.

Hardware-based RAID arrays are transparent to the operating system, which makes their management easier. Software RAID implementations have several configuration options, tending to complicate management.

In contrast, software-based RAID systems are extremely cost efficient (since cheaper IDE disks can be used without any need for additional hardware). With software-based systems, different partitions in a single disk can be grouped together to create redundancy and fault-tolerance, whereas hardware-based systems group together whole drives. Also, under Linux, software-based RAID management is completely possible within the operating system, whereas dedicated management tools under Linux for hardware implementations may not be available.

(C) Installing and configuring RAID in Linux

(D) Supported controllers

Currently a well-supported host-based hardware RAID controller (i.e,

a controller for which there exists one or more well-supported drivers under Linux) is one that is made by DPT (http://www.dpt.com). However, there exist other host-based and SCSI-to-SCSI controllers which may work under Linux. These include the ones made by Syred (http://www.syred.com), ICP-Vortex (http://www.icp-vortex.com), and BusLogic (http://www.mylex.com). See the RAID solutions for Linux page http://linas.org/linux/raid.html for more information.

(E) DPT controllers

Essentially all the SmartRAID IV and V controllers are supported. The installation and configuration section will focus on how to set up SCSI-based hardware RAID, focusing mainly on host-based adapters from DPT, though the principles applied here are fairly general. Specifically, the Smartcache IV PM2144UW and PM3334UW controllers have been tested, with DPT supplied enclosures, on Linux 2.0.3x and 2.2.x kernels. The SmartRAID V Decade cards require additional software. For additional detail and the latest information, consult the DPT Hardware RAID HOWTO at http://www.ram.org/computing/linux/dpt_raid.html.

(E) ICP vortex controllers

ICP vortex has a complete line of disk array controllers which support Linux. The ICP driver is in the Linux kernel since version 2.0.31. All major Linux Distributors S.u.S.e., LST Power Linux, Caldera and Red Hat support the ICP controllers as boot/installation controllers. The RAID system can easily be configured with their romsetup program (you do not have to boot MS-DOS for configuration). With their monitoring utility gdtmon it is possible to manage the complete ICP RAID system during operation (check transfer rates, set parameters for the controller and hard disks, exchange defective hard disks,etc.).

(D) What hardware should be used?

(E) Controller type

Given all these options, if you're looking for a RAID solution, you need to think carefully about what you want. Depending on what you want to do, and which RAID level you wish to use, some cards may be better than others. SCSI-to-SCSI adapters may not be as good as host- based adapters, for example. See the discussion by Michael Neuffer (neuffer@kralle.zdv.uni-mainz.de), the author of the EATA-DMA driver, on his Linux High Performance SCSI and RAID page

http://www.uni-mainz.de/ neuffer/scsi/.

(E) Enclosure type

The enclosure type affects the hot swap-ability of the drive, the warning systems (i.e., whether there will be indication of failure, and whether you will know which drive has failed), and what kind of treatment your drive receives (for example, redundant cooling and power supplies). We used the DPT supplied enclosures which have worked extremely well, but they are expensive. We have also used Wetex enclosures (http://www.wetex.com) and they also work fairly well, but lacked the hot swap-ability of the DPT enclosures.

(D) Installation

(E) Installing and configuring the hardware

Refer to the instruction manual to install the controller and the drives. For DPT, since a storage manager for Linux doesn't exist yet, you need to create a MS-DOS-formatted disk with the system on it (usually created using the command format /s at the MS-DOS prompt). You will also be using the DPT storage manager for MS-DOS, which you should probably make a copy of for safety.

Once the hardware is in place, boot using the DOS system disk. Replace the DOS disk with the storage manager. And invoke the storage manager using the command:

```
a:\ dptmgr
```

Wait a minute or so, and you'll get a nice menu of options. Configure the set of disks as a hardware RAID (single logical array). Choose "other" as the operating system.

The MS-DOS storage manager is a lot easier to use with a mouse, and so you might want to have a mouse driver on the initial system disk you create.

Technically, it should be possible to run the SCO storage manager under Linux, but it may be more trouble than its worth. It's probably more easier to run the MS-DOS storage manager under Linux.

(E) Configuring the kernel

You will need to configure the kernel with SCSI support and the appropriate low level driver. See the Kernel HOWTO

http://sunsite.unc.edu/mdw/HOWTO/Kernel-HOWTO.html for information on how to compile the kernel. Once you choose "yes" for SCSI support, in the low level drivers section, select the driver of your choice (EATA DMA or EATA ISA/EISA/PCI for most EATA DMA compliant (DPT) cards, EATA PIO for the very old PM2001 and PM2012A from DPT). Most drivers, including the EATA DMA and EATA ISA/EISA/PCI drivers, should be available in recent kernel versions.

Once you have the kernel compiled, reboot, and if you've set up everything correctly, you should see the driver recognising the RAID as a single SCSI disk. If you use RAID-5, you will see the size of this disk to be 2/3 of the actual disk space available.

(E) Bootup messages

The messages you see upon bootup if you're using the EATA DMA driver should look something like this:

```
EATA (Extended Attachment) driver version: 2.59b
developed in co-operation with DPT
(c) 1993-96 Michael Neuffer, mike@i-Connect.Net
Registered HBAs:
HBA no. Boardtype    Revis  EATA Bus  BaseIO IRQ DMA Ch ID Pr QS  S/G IS
scsi0 : PM3334UW     v07L.0 2.0c PCI  0xef90  10 BMST 3  7  N  64 252 Y
scsi1 : PM2144UW     v07L.Y 2.0c PCI  0xef50   9 BMST 1  7  N  64 252 Y
scsi2 : PM2044U      v07K.V 2.0c PCI  0xef10  11 BMST 1  7  N  64 252 Y
scsi0 : EATA (Extended Attachment) HBA driver
scsi1 : EATA (Extended Attachment) HBA driver
scsi2 : EATA (Extended Attachment) HBA driver
scsi : 3 hosts.
  Vendor: DPT       Model: RAID-5            Rev: 07L0
  Type:   Direct-Access                      ANSI SCSI revision: 02
Detected scsi disk sda at scsi0, channel 0, id 0, lun 0
scsi0: queue depth for target 0 on channel 0 set to 64
  Vendor: DPT       Model: ATOS              Rev: 07LY
  Type:   Direct-Access                      ANSI SCSI revision: 02
Detected scsi disk sdb at scsi1, channel 0, id 8, lun 0
scsi1: queue depth for target 8 on channel 0 set to 64
  Vendor: SONY      Model: SDT-9000          Rev: 0200
  Type:   Sequential-Access                  ANSI SCSI revision: 02
scsi2: queue depth for target 0 on channel 0 set to 64
SCSI device sda: hdwr sector= 512 bytes. Sectors= 105676800 [51600 MB] [51.6 GB]
 sda: sda1
SCSI device sdb: hdwr sector= 512 bytes. Sectors= 35561984 [17364 MB] [17.4 GB]
 sdb: sdb1
```

The above display is for a setup with three DPT SCSI controllers, with two RAID-5 arrays of size 51.6 GB (seven 9 GB disks) and 17.4 GB (three 9 GB disks).

The messages you see upon bootup if you're using the EATA ISA/EISA/PCI driver should look something like this:

```
EATA0: IRQ 11 mapped to IO-APIC IRQ 16.
EATA/DMA 2.0x: Copyright (C) 1994-1999 Dario Ballabio.
EATA config options -> tc:y, lc:n, mq:16, eh:y, rs:y, et:n.
EATA0: 2.0C, PCI 0xef10, IRQ 16, BMST, SG 122, MB 64.
EATA0: SCSI channel 0 enabled, host target ID 7.
EATA1: IRQ 9 mapped to IO-APIC IRQ 18.
EATA1: 2.0C, PCI 0xef50, IRQ 18, BMST, SG 122, MB 64.
EATA1: wide SCSI support enabled, max_id 16, max_lun 8.
EATA1: SCSI channel 0 enabled, host target ID 7.
scsi0 : EATA/DMA 2.0x rev. 5.10.00
scsi1 : EATA/DMA 2.0x rev. 5.10.00
scsi : 2 hosts.
  Vendor: SONY      Model: SDT-9000          Rev: 0200
  Type:   Sequential-Access                  ANSI SCSI revision: 02
EATA0: scsi0, channel 0, id 0, lun 0, cmds/lun 2.
  Vendor: DPT       Model: ATOS              Rev: 07LY
  Type:   Direct-Access                      ANSI SCSI revision: 02
Detected scsi disk sda at scsi1, channel 0, id 8, lun 0
EATA1: scsi1, channel 0, id 8, lun 0, cmds/lun 16, unsorted, tagged.
SCSI device sda: hdwr sector= 512 bytes. Sectors= 35561984 [17364 MB] [17.4 GB]
 sda: sda1
```

The above display is for a setup with two DPT SCSI controllers, with one RAID-5 array of size 17.4 GB (three 9 GB disks).

(D) Configuring and testing

(E) Setting up the array

You can now start treating the RAID as a regular disk. The first thing you'll need to do is partition the disk (using fdisk). You'll then need to set up an ext2 filesystem. This can be done by running the command:

```
# mkfs -t ext2 /dev/sdxN
```

where /dev/sdxN is the name of the SCSI partition. Once you do this, you'll be able to mount the partitions and use them as you would any other disk (including adding entries in /etc/fstab).

(E) Hot swapping

We first tried to test hot swapping by removing a drive and putting it back in the DPT-supplied enclosure/tower (which you buy for an

additional cost). Before we could carry this out to completion, one of the disks failed. Even though one of the disks failed, all the data on the RAID drive was accessible.

Instead of replacing the drive, we just went through the motions of hot swapping and put the same drive back in. The drive rebuilt itself and everything turned out okay. During the time the disk had failed, and during the rebuilding process, all the data was accessible. Though it should be noted that if another disk had failed, we'd have been in serious trouble.

(E) Performance

Here are some timings with a 3344 UW controller using the bonnie program:

```
       -------Sequential Output-------- ---Sequential Input-- --Random--
       -Per Char- --Block--- -Rewrite-- -Per Char- --Block--- --Seeks---
    MB K/sec %CPU K/sec %CPU K/sec %CPU K/sec %CPU K/sec %CPU  /sec %CPU
  1000  1714 17.2  1689  6.0  1200  5.7  5263 40.2  7023 12.1  51.3 2.2
```

(E) Features in the EATA DMA driver

This section describes some of the commands available under Linux to check on the RAID configuration. Again, while references to the eata_dma driver is made, this can be used to check up on any driver.

To see the configuration for your driver, type:

```
# cat /proc/scsi/eata_dma/N
```

where N is the host id for the controller. You should see something like this:

```
EATA (Extended Attachment) driver version: 2.59b
queued commands:          17664
processed interrupts:     17664

scsi0 : HBA PM3334UW
Firmware revision: v07L.0
Hardware Configuration:
IRQ:  0, edge triggered
DMA: BUSMASTER
CPU: MC68000 0MHz
Base IO : 0xef90
```

```
Host Bus: PCI
SCSI Bus: Speed: 5MB/sec.
SCSI channel expansion Module: not present
SmartRAID hardware: not present.
    Type: -
    Max array groups:              7
    Max drives per RAID 0 array:   7
    Max drives per RAID 3/5 array: 7
Cache Module: not present.
    Type: -
    Bank0: 0MB without ECC
    Bank1: 0MB without ECC
    Bank2: 0MB without ECC
    Bank3: 0MB without ECC
Timer Mod.: not present
NVRAM     : not present
SmartROM  : enabled
Alarm     : off
Host<->Disk command statistics:
         Reads:      Writes:
    1k:           0            0
    2k:           0            0
    4k:           0            0
    8k:           0            0
   16k:           0            0
   32k:           0            0
   64k:           0            0
  128k:           0            0
  256k:           0            0
  512k:           0            0
 1024k:           0            0
>1024k:           0            0
Sum   :           0            0
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
  Vendor: DPT      Model: RAID-5           Rev: 07L0
  Type:   Direct-Access                    ANSI SCSI revision: 02
```

To get advanced command statistics, type:

```
# echo "eata_dma latency" > /proc/scsi/eata_dma/N
```

Then you can do a:

```
# cat /proc/scsi/eata_dma/N
```

to get more detailed statistics:

```
EATA (Extended Attachment) driver version: 2.59b
queued commands:          17737
processed interrupts:     17737

scsi0 : HBA PM3334UW
Firmware revision: v07L.0
Hardware Configuration:
IRQ:  0, edge triggered
DMA: BUSMASTER
CPU: MC68000 0MHz
Base IO : 0xef90
Host Bus: PCI
SCSI Bus: Speed: 5MB/sec.
SCSI channel expansion Module: not present
SmartRAID hardware: not present.
    Type: -
    Max array groups:              7
    Max drives per RAID 0 array:   7
    Max drives per RAID 3/5 array: 7
Cache Module: not present.
    Type: -
    Bank0: 0MB without ECC
    Bank1: 0MB without ECC
    Bank2: 0MB without ECC
    Bank3: 0MB without ECC
Timer Mod.: not present
NVRAM     : not present
SmartROM  : enabled
Alarm     : off
Host Latency Command Statistics:
Current timer resolution: 10ms
         Reads:       Min:(ms)     Max:(ms)     Ave:(ms)
    1k:          36            0           30            9
    2k:           0            0            0            0
    4k:           3            0           10            6
    8k:           0            0            0            0
   16k:           0            0            0            0
   32k:           0            0            0            0
   64k:           0            0            0            0
  128k:           0            0            0            0
  256k:           0            0            0            0
  512k:           0            0            0            0
 1024k:           0            0            0            0
>1024k:           0            0            0            0
         Writes:      Min:(ms)     Max:(ms)     Ave:(ms)
    1k:          24            0            0            0
    2k:           0            0            0            0
    4k:           0            0            0            0
    8k:           0            0            0            0
   16k:           0            0            0            0
   32k:           0            0            0            0
   64k:           0            0            0            0
  128k:           0            0            0            0
```

```
  256k:           0            0            0            0
  512k:           0            0            0            0
 1024k:           0            0            0            0
>1024k:           0            0            0            0
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
  Vendor: DPT      Model: RAID-5           Rev: 07L0
  Type:   Direct-Access                    ANSI SCSI revision: 02
```

To turn off advanced command statistics, type:

```
# echo "eata_dma nolatency" > /proc/scsi/eata_dma/N
```

(C) Troubleshooting

(D) Upon bootup, no SCSI hosts are detected

This could be due to several reasons, but it's probably because the appropriate driver is not configured in the kernel. Check and make sure the appropriate driver (EATA-DMA or EATA ISA/EISA/PCI for most DPT cards) is configured and installed. Type in the command lsmod to see that the driver is indeed loaded and check the messages produced by syslog.

(D) RAID configuration shows up as N different disks

The RAID has not been configured properly. If you're using a DPT storage manager, you need to configure the RAID disks as a single logical array. When you configure the controller with the SM start it with the parameter /FW0 and/or select Solaris as OS. This will cause the array setup to be managed internally by the controller.

(D) Machine/controller is shut down in the middle of a format

As stated in the DPT manual, this is clearly a no-no and might require the disks to be returned to the manufacturer, since the DPT Storage Manager might not be able format it. However, you might be able to perform a low level format on it, using a program supplied by DPT, called clfmt in their utilities page http://www.dpt.com/techsup/. Read the instructions after unzipping the clfmt.zip file on how to use it (and use it wisely). Once you do the low level format, you might be able to treat the disks like new. Use this program carefully!

(D) SCSI_ABORT_BUSY errors produced during initial filesystem format

When you do a mke2fs on the SCSI drive, you may see errors of the form:

```
scsi: aborting command due to timeout : pid xxx, scsi0, channel 0, id 2, lun 0
write (10) xx xx xx xx xx xx xx xx xx
eata_abort called pid xxx target: 2 lun: 0 reason: 3
Returning: SCSI_ABORT_BUSY
```

and this might end up causing the machine to freeze. I (and many others) have been able to fix this problem by simply reading one or two hundred MB from the RAID array with dd like this:

```
# dd if=/dev/sdX of=/dev/null bs=1024k count=128
```

During a format, a fast rush of requests for chunks of memory that is directly accessible is made, and sometimes the memory manager cannot deliver it on time anymore. The dd is a workaround that will simply create the requests sequentially instead of one huge heap at once like the format tends to create it.

(D) If all fails...

Read this chapter and all the referenced URLs again. Check the cabling and the termination. Try a different machine if you have access to one. The most common cause of problems with SCSI devices and drivers is because of faulty or misconfigured hardware. Finally, you can post to the various newsgroups and ask for help.