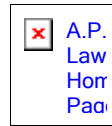


Search
FAQ
New to Sco
New to Unix
New Here
SCO News
Unix Articles
Unixware
Linux
Security
Code
Tests
Links
Books
Reviews
Opinion
Find a Consultant
Find a Job
Site Forum
Donations
Contact Info
Services
Sales
Consulting Rates
Advertising Rates
Site Map



A.P. Lawrence Home

Over 10,600 Unix/Linux users so far this month (48,500+ page views)

 <http://www.microlite.com>

[More Articles](#)

Raid

Just a very few years ago, RAID was an expensive option. That's all changed, and today anyone with high disk performance needs or concerns about data reliability should consider some sort of RAID configuration.

RAID means "Redundant Array of Inexpensive Disks". There are basically 5 defined levels of RAID:

RAID 0

Striped disks. Highest performance, but no redundancy, and therefore really isn't RAID at all. Very seldom used because of the reduced reliability: if one drive fails, the entire array fails.

However, it is the concept of striping that is important to understand: rather than writing a data to sequential blocks on one drive, each subsequent block is written to the next physical drive. This gives great read performance because multiple drives work on getting their portion of the data, thus delivering it back to the computer much faster than one drive alone could.

To the OS and to the user, a striped RAID drive looks like ONE larger drive. Indeed, in hardware RAID implementations, you wouldn't know at all that this was not just one big disk without special software provided by the RAID manufacturer. Note that there is nothing special about the disk drives themselves: it is the disk controller that provides the abstraction that makes the striped drives look like one drive.

- RAID 1

Mirrored disks. Each drive has a twin and all data is written to both drives, and can be read from either drive. This increases read performance and, if accomplished by hardware rather than software (see below) has no adverse affect on write performance. There's no striping here; the read gain is only due to the fact that the controller can read from whichever drive is less busy or whose heads are closer to the data. In the event of one drive failing, the twin drive is immediately available with little performance

degradation. When the failed drive is replaced, the new drive is rebuilt by reading data from the good drive.

On very high end configurations, the twin can also provide a "snap-shot" ability, where writes to one of the twins are temporarily turned off so that a backup can be done as of that moment in time while continuing to allow writes to the other drive. This feature is often found in software configurations (see [Veritas Volume Manager](#)) or high end hardware/software combinations, but not in the inexpensive segment of the market.

In both hardware and software implementations, it is possible to have more than one mirror for data that is really critical. While expensive, this provides even greater security against catastrophic failure and (with properly designed software/firmware) continues to increase read performance for each mirror added.

This mirroring can be combined with RAID 0, giving striped disks that are mirrored. Although this offers the high performance of RAID 0 and redundancy, it is much more expensive and typically not seen except in environments where cost just doesn't matter.

- RAID 2

Seldom implemented. This is striping, but with some drives storing ECC data. As all modern drives implement ECC information themselves, there's no particular advantage to this configuration.

- RAID 3

Stripes data across multiple drives, but dedicates one drive to parity information. Data is read from all drives at once. This uses the imbedded ECC data to detect errors, and recovers data using the parity information. RAID 3 can give high performance in dedicated situations where large amounts of data need to be read quickly. It requires synchronized spindle drives and really doesn't perform well in multi-user situations and therefore is not usually seen except in unusual and very specialized circumstances.

The concept of parity utilizes the mathematical properties of the XOR (Exclusive OR). You can see how this works by using the [Javascript Bit Twiddler](#). For example, if you XOR the values 12 and 15, you'll get 3. If you XOR 3 (the result of your first XOR) with either 12 or 15, you'll get the other value. That "3" would be the parity information that would be used to reconstruct the "12" or the "15" if each of those represented data stored on a different disk. The XOR is a very quick operation, handled directly by the CPU. but of course it does involve some overhead, so writing involves both an extra calculation (the XOR) and another disk write (the disk write is concurrent with the other writes though, so that really doesn't hurt anything).

If a drive fails, the controller provides the "missing" data by calculating it from the data it does have: the other data drive(s) and the parity drive. This is, of course, slower than reading it from the disk would be.

This also suffers badly on write performance because the parity drive will need to be written constantly, making it impossible to overlap multiple writes.

- RAID 4

Very similar to RAID 3, but allows individual drives to be separately read. It has no particular advantage over RAID 5 and the disadvantage of a dedicated parity drive.

- RAID 5

This is a popular configuration offering excellent read performance and high reliability. The concept here is to have parity data, but to spread it over all the drives. This lets writes overlap because typical small writes access one data drive and one parity drive. If another write is accessing a different set of drives, the two writes can be done in parallel, which is not possible with a dedicated parity drive as described above. This requires a minimum of 3 disk drives, and more is better. RAID 5 is less expensive than mirroring (for equivalent storage), can provide very fast reads, particularly with more than 3 drives, and can survive a single drive failure. The disadvantage is that write performance is not as good (but most applications do much more reading than writing) and that in the event of failure, both read and write performance suffer due to the overhead involved in reconstructing data from parity information.

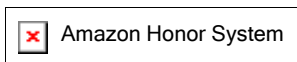
Recently, non-official designations such as RAID 6 have been offered at the very high end of the market. These are really just RAID 5 implementations but with multiple parity writes, so that the array can withstand the failure of multiple drives simultaneously. Obviously only very high end systems need such redundancy.

As alluded to above, RAID can be implemented in hardware or software. There can also be configurations that are really both: Sun's high end RAID products are tightly coupled software and hardware.

It used to be that RAID was always SCSI based. That's no longer true; inexpensive IDE RAID configurations are now available. They are not going to have the performance characteristics of a SCSI design, but they cost less, and certainly would give good value for the money.

Was this information valuable to you? Consider making a [contribution](#) to help pay for the costs of bringing it to you.

[Amazon Honor System](#) it's fast, safe and easy. It's also very private- Amazon doesn't share your information with anyone: I don't even get to find out who contributed!



Related Articles:

[SCO Unix/OSR5 Technical FAQ](#) (Basics)

[Shell Bashing](#) (Administration, Basics, Linux)

[VPN's and other remote access](#) (Basics, Linux, Security, Networking)

[Unix Permissions](#) (Basics, Linux)

[Transferring Data from an Old Hard Drive](#) (Disks/Filesystems)

[Basic Scripting](#) (Basics, Programming)

[Adding a Hard Drive to Linux](#) (Linux, Basics, Disks/Filesystems)

[Hard Disks](#) (Basics, Disks/Filesystems)

[Loose Cable](#) (Disks/Filesystems)

[SCO Unix/OSR5 Installation FAQ](#) (Basics)
[Floppies](#) (Basics, Disks/Filesystems)
[Basics](#) (Basics)
[General Trouble Shooting](#) (Basics, Administration)
[Cron, At and Batch](#) (Kernel/Internals, Basics)
[Bootting OSR5- Filesystems](#) (Kernel/Internals, Disks/Filesystems)
[Bootting OSR5](#) (Kernel/Internals, Disks/Filesystems)
[Bootting OSR5- Definitions](#) (Kernel/Internals, Disks/Filesystems)
[Bootting OSR5- Swap and Dump](#) (Kernel/Internals, Disks/Filesystems)
[Understanding Termcap and Terminfo](#) (Basics, Modems/Terminals)
[Understanding SCSI](#) (Disks/Filesystems, Basics)
[DPT Raid Controller](#) (Reviews, Disks/Filesystems)
[Understanding Dumb Terminals](#) (Basics, Modems/Terminals)
[Veritas \(Online Data Manager\)](#) (Unixware, Disks/Filesystems)
[Using Tapes and tape drives](#) (Basics, Backup)
[Configuring SCO's GUI](#) (Misc., Administration, Basics)
[Networks 101](#) (Networking, Basics)
[Using Secondary drives](#) (Administration, Disks/Filesystems)
[OSR5 Boot Up Messages](#) (Kernel/Internals, Basics)
[Out of Disk space](#) (Disks/Filesystems)
[Adding another hard drive](#) (Administration, Disks/Filesystems)
[Using the Korn Shell](#) (Programming, Basics)
[Vi Primer](#) (Basics)
[Routing Basics](#) (Networking, Basics)
[Understanding Serial Wiring](#) (Basics, Modems/Terminals)
[bfind.c- Finds which file contains a block](#) (Programming, Disks/Filesystems, Kernel/Internals, Code)

© March 1999 Anthony Lawrence. All rights reserved.

This article is copyrighted material. You have permission to use it for any purpose, commercial or non-commercial, as long as it is kept intact and no modifications, additions, or deletions are made except as allowed herein.

You may publish it in paper or electronic form. That includes magazine, newsletters, and web pages, both internal and external, for profit or not. Banner ads and other graphics may be removed, but all other text, hyperlinks and copyright notices, including this, must remain (but see below also). You may not delete text, alter it, or add to it in any way that does not clearly delineate what is yours and what comes from this site. You may alter fonts, font sizes and the like and reformat text as is appropriate for your use.

You may select specific paragraphs or sections, but if you do so, you must include this entire notice also, noting that you have not published the entire article, or simply note that the paragraphs you have published are part of a larger article and give the http address of the actual article.

My main concern is that no one would be confused that you wrote something I wrote or vice-versa. If your use meets that concern, and allows people to find the original article here, I have no objection.

This general permission specifically does NOT apply to test questions and answers.

Some articles at <http://www.aplawrence.com/> and <http://www.pcunix.com/> are copyrighted by other individuals or corporations; these paragraphs do not apply to those articles even if accidentally included.

We do appreciate being advised of any such use: Email: tony@aplarence.com.