Software-RAID HOWTO

Linas Vepstas, linas@linas.org v0.54, 21 November 1998

최 희철 ironyjk@kldp.org 2000년 3월 1일

RAID 는 "Redundant Array of Inexpensive Disks" 의 약자로, 각각의 디스크 들을 묶어서 빠르고 안전한 디스크 시스템을 만드는 것이다. RAID 는 하나의 디스크에 비해 오류를 대비할 수 있으며, 속도를 증가 시킨다.

RAID stands for "Redundant Array of Inexpensive Disks", and is meant to be a way of creating a fast and reliable disk-drive subsystem out of individual disks. RAID can guard against disk failure, and can also improve performance over that of a single disk drive.

이 문서는 Linux MD kernel 확장에 관한 tutorial/HOWTO/FAQ 문서이다. MD 확장은 RAID-0,1,4,5 를 소프트웨어 적으로 지원하고, 이것을 통해 우리는 특별한 하드웨어나 디스크 콘트롤러 없이 RAID 를 사용해 볼 수 있다.

This document is a tutorial/HOWTO/FAQ for users of the Linux MD kernel extension, the associated tools, and their use. The MD extension implements RAID-0 (striping), RAID-1 (mirroring), RAID-4 and RAID-5 in software. That is, with MD, no special hardware or disk controllers are required to get many of the benefits of RAID.

목 록

RAID는 디스크의 추가로 시스템의 신뢰성을 향상시킬수 있으나, 잘못된 사용으로 인해 역효과를 낼수도 있다. 특히 , RAID는 디스크 자체의 오류에 대비한 것이지. 사용자의 실수나, 전원의 불량에 대비하도록 설계된 것이 아니다. 전원의 불량과, 개발 커널, 그리고, 관리자의 실수는 데이터를 손상시킬 것이고, RAID 는 백업방법이 아님에 유의하라.

RAID, although designed to improve system reliability by adding redundancy, can also lead to a false sense of security and confidence when used improperly. This false confidence can lead to even greater disasters. In particular, note that RAID is designed to protect against *disk* failures, and not against *power* failures or *operator* mistakes. Power failures, buggy development kernels, or operator/admin errors can lead to damaged data that it is not recoverable! RAID is *not* a substitute for proper backup of your system. Know what you are doing, test, be knowledgeable and aware!

1 소개

1. **Q**: RAID 란 무엇인가?

   A

   RAID 는 "Redundant Array of Inexpensive Disks' 의 약자로, 각각의 디스크 들을 묶어서 빠르고 안전한 디스크 시스템을 만드는 것이다.

   RAID stands for "Redundant Array of Inexpensive Disks", and is meant to be a way of creating a fast and reliable disk-drive subsystem out of individual disks. In the PC world, "I" has come to stand for "Independent", where marketing forces continue to differentiate IDE and SCSI. In it's original meaning, "I" meant "Inexpensive as compared to refrigerator-sized mainframe 3380 DASD", monster drives which made nice houses look cheap, and diamond rings look like trinkets.

2. **Q**: 이 문서는 무엇인가?

   A:

   이 문서는 Linux MD kernel 확장에 관한 tutorial/HOWTO/FAQ 문서이다. MD 확장은 RAID-0,1,4,5 를 소프트웨어 적으로 지원하고, 이것을 통해 우리는 특별한 하드웨어나 디스크 콘트롤러 없이 RAID 를 사용해 볼수 있다.

   This document is a tutorial/HOWTO/FAQ for users of the Linux MD kernel extension, the associated tools, and their use. The MD extension implements RAID-0 (striping), RAID-1 (mirroring), RAID-4 and RAID-5 in software. That is, with MD, no special hardware or disk controllers are required to get many of the benefits of RAID.

   This document is **NOT** an introduction to RAID; you must find this elsewhere.

3. **Q**: Linux 커널은 어떤 레벨의 RAID 를 지원하는가?

A: RAID-0 은 2.x 버전의 리눅스 커널들이 지원한다. 이것은 이해하기 쉽고, 몇몇의 매우 큰 유즈넷 뉴스 서버에 사용된다.

Striping (RAID-0) and linear concatenation are a part of the stock 2.x series of kernels. This code is of production quality; it is well understood and well maintained. It is being used in some very large USENET news servers.

RAID-1, RAID-4, RAID-5 는 커널 2.1.63 이상의 버전에서 지원한다. 2.0.x 와 2.1.x의 커널들은 패치를 해야 한다. 커널을 업그레이드 해야 한다고 생각하지 마라. 업그레이드보다. 패치가 훨씬 쉬울 것이다.

RAID-1, RAID-4 & RAID-5 are a part of the 2.1.63 and greater kernels. For earlier 2.0.x and 2.1.x kernels, patches exist that will provide this function. Don't feel obligated to upgrade to 2.1.63; upgrading the kernel is hard; it is *much* easier to patch an earlier kernel. Most of the RAID user community is running 2.0.x kernels, and that's where most of the historic RAID development has focused. The current snapshots should be considered near-production quality; that is, there are no known bugs but there are some rough edges and untested system setups. There are a large number of people using Software RAID in a production environment.

RAID-1 hot reconstruction has been recently introduced (August 1997) and should be considered alpha quality. RAID-5 hot reconstruction will be alpha quality any day now.

A word of caution about the 2.1.x development kernels: these are less than stable in a variety of ways. Some of the newer disk controllers (e.g. the Promise Ultra's) are supported only in the 2.1.x kernels. However, the 2.1.x kernels have seen frequent changes in the block device driver, in the DMA and interrupt code, in the PCI, IDE and SCSI code, and in the disk controller drivers. The combination of these factors, coupled to cheapo hard drives and/or low-quality ribbon cables can lead to considerable heartbreak. The `ckraid` tool, as well as `fsck` and `mount` put considerable stress on the RAID subsystem. This can lead to hard lockups during boot, where even the magic alt-SysReq key sequence won't save the day. Use caution with the 2.1.x kernels, and expect trouble. Or stick to the 2.0.34 kernel.

4. **Q**: 어디에서 커널의 패치를 구할 수 있나요?

A: Software RAID-0 and linear mode are a stock part of all current Linux kernels. Patches for Software RAID-1,4,5 are available from `<http://luthien.nuclecu.unam.mx/miguel/raid>`. See also the quasi-mirror `<ftp://linux.kernel.org/pub/linux/daemons/raid/>` for patches, tools and other goodies.

5. **Q**: Linux RAID 에 관한 다른 문서들이 있나요?

A:

- Generic RAID overview: `<http://www.dpt.com/uraiddoc.html>`.
- General Linux RAID options: `<http://linas.org/linux/raid.html>`.
- Latest version of this document: `<http://linas.org/linux/Software-RAID/Software-RAID.html>`.
- Linux-RAID mailing list archive: `<http://www.linuxhq.com/lnxlists/>`.
- Linux Software RAID Home Page: `<http://luthien.nuclecu.unam.mx/miguel/raid>`.
- Linux Software RAID tools: `<ftp://linux.kernel.org/pub/linux/daemons/raid/>`.
- How to setting up linear/stripped Software RAID: `<http://www.ssc.com/lg/issue17/raid.html>`.
- Bootable RAID mini-HOWTO: `<ftp://ftp.bizsystems.com/pub/raid/bootable-raid>`.
- Root RAID HOWTO: `<ftp://ftp.bizsystems.com/pub/raid/Root-RAID-HOWTO>`.

- Linux RAID-Geschichten: <http://www.infodrom.north.de/joey/Linux/raid/>.

6. **Q**: 이 문서에 대해 누구에게 불평해야 하죠?

   A: Linas Vepstas slapped this thing together. However, most of the information, and some of the words were supplied by

   - Bradley Ward Allen <ulmo@Q.Net>
   - Luca Berra <bluca@comedia.it>
   - Brian Candler <B.Candler@pobox.com>
   - Bohumil Chalupa <bochal@apollo.karlov.mff.cuni.cz>
   - Rob Hagopian <hagopiar@vu.union.edu>
   - Anton Hristozov <anton@intransco.com>
   - Miguel de Icaza <miguel@luthien.nuclecu.unam.mx>
   - Marco Meloni <tonno@stud.unipg.it>
   - Ingo Molnar <mingo@pc7537.hil.siemens.at>
   - Alvin Oga <alvin@planet.fef.com>
   - Gadi Oxman <gadio@netvision.net.il>
   - Vaughan Pratt <pratt@cs.Stanford.EDU>
   - Steven A. Reisman <sar@pressenter.com>
   - Michael Robinton <michael@bzs.org>
   - Martin Schulze <joey@finlandia.infodrom.north.de>
   - Geoff Thompson <geofft@cs.waikato.ac.nz>
   - Edward Welbon <welbon@bga.com>
   - Rod Wilkens <rwilkens@border.net>
   - Johan Wiltink <j.m.wiltink@pi.net>
   - Leonard N. Zubkoff <lnz@dandelion.com>
   - Marc ZYNGIER <zyngier@ufr-info-p7.ibp.fr>

   Copyrights

   Thanks all for being there!

2 RAID 의 이해

1. **Q**: RAID는 무엇인가? 왜 나는 아직 사용해 보지 않았는가?

   A: RAID 는 여러개의 디스크를 엮어서, 속도와, 안전성이 좋은 하나의 형태로 만드는 것이다. RAID 는 여러가지 형태가 있고, 그 형태마다 각각의 장단점을 가지고 있다. 예를 들면 RAID 레벨 1 은 두개(혹은 이상)의 디스크에 같은 데이터의 복사본을 넣는 것이다. 데이터가 복사된 각 디스크에서 데이터를 가져오기 때문에 읽는 속도는 빨라질 것이다. 추가적으로 복사된 데이터는 하나의 디스크가 깨졌을 때 안정성을 제공할 것이다. RAID 레벨에 의한 다른 방법은, 여러개의 디스크를 하나의 디스크로 묶는 것이다. 그것은 간단한 복사에 비해 좀 더 많은 저장률을 제공할 것이다, 또한, 읽기 쓰기를 위한 성능 향상을 시키면서, 여전히 오류에 대비한 적당한 여유공간을 남겨둘 것이다.

RAID is a way of combining multiple disk drives into a single entity to improve performance and/or reliability. There are a variety of different types and implementations of RAID, each with its own advantages and disadvantages. For example, by putting a copy of the same data on two disks (called **disk mirroring**, or RAID level 1), read performance can be improved by reading alternately from each disk in the mirror. On average, each disk is less busy, as it is handling only 1/2 the reads (for two disks), or 1/3 (for three disks), etc. In addition, a mirror can improve reliability: if one disk fails, the other disk(s) have a copy of the data. Different ways of combining the disks into one, referred to as **RAID levels**, can provide greater storage efficiency than simple mirroring, or can alter latency (access-time) performance, or throughput (transfer rate) performance, for reading or writing, while still retaining redundancy that is useful for guarding against failures.

RAID는 디스크 오류에 대비할 수 있지만, 인간의 실수나, 프로그램의 오류에는 대비할 수 없다. (RAID 프로그램 자체도 오류를 포함할 수 있다.) net 상에는 RAID 설치에 익숙치 않은 관리자들이 그들의 데이터를 모두 잃어버리는 그런 비극적인 이야기 들이 많다. RAID는 백업의 대체 할 수 없음에 주의하라!

Although RAID can protect against disk failure, it does not protect against operator and administrator (human) error, or against loss due to programming bugs (possibly due to bugs in the RAID software itself). The net abounds with tragic tales of system administrators who have bungled a RAID installation, and have lost all of their data. RAID is not a substitute for frequent, regularly scheduled backup.

RAID 는 하드웨어 적으로, 특별한 디스크 콘트롤러로, 또는, 커널 모듈 같은 소프트웨어 적으로 구현될 수 있다. RAID 하드웨어 "디스크 콘트롤러"는 디스크 드라이브에 케이블을 연결할수 있게 해주는 것이다. 일반적으로 lISA/EISA/PCI/S-Bus/MicroChannel 슬롯에 장착할수 있는 카드형식이나, 어떤 것들은 일반적인 컨트롤러와 디스크사이를 연결해는 박스 형식이다.

RAID can be implemented in hardware, in the form of special disk controllers, or in software, as a kernel module that is layered in between the low-level disk driver, and the file system which sits above it. RAID hardware is always a "disk controller", that is, a device to which one can cable up the disk drives. Usually it comes in the form of an adapter card that will plug into a ISA/EISA/PCI/S-Bus/MicroChannel slot. However, some RAID controllers are in the form of a box that connects into the cable in between the usual system disk controller, and the disk drives. Small ones may fit into a drive bay; large ones may be built into a storage cabinet with its own drive bays and power supply.

최신의 빠른 CPU를 사용하는 RAID 하드웨어는 최고의 속도를 내긴 하지만, 그만큼 비쌀 것이다. 이유는 대부분이 보드상에 충분한 DSP 와 메모리를 가지고 있기 때문이다. 오래된 RAID 하드웨어는 DSP와 캐쉬의 병목현상으로 최신의 CPU를 사용하는 시스템의 속도를 저하시킬수 있다. 때로는, 일반적인 하드웨어와 소프트웨어 RAID 를 사용하는 것보다 더 느릴 것이다. 하드웨어 RAID가 소프트웨어 RAID에 비해 장점이 있을 수 있지만, 최근 대부분의 디스크 드라이브들에겐 그렇지 않다.? RAID 하드웨어는 일반적으로 다른 메이커와 모델의 하드들에게 호환성을 제공하지 않지만, 리눅스상의 소프트웨어 RAID는 어떤 특별한 설정없이 대부분의 하드웨어들이 잘 돌아갈 것이다.

The latest RAID hardware used with the latest & fastest CPU will usually provide the best overall performance, although at a significant price. This is because most RAID controllers come with on-board DSP's and memory cache that can off-load a considerable amount of processing from the main CPU, as well as allow high transfer rates into the large controller cache. Old RAID hardware can act as a "de-accelerator" when used with newer CPU's: yesterday's fancy DSP and cache can act as a bottleneck, and it's performance is often beaten by pure-software RAID and new but otherwise plain, run-of-the-mill disk controllers. RAID hardware can offer an advantage over pure-software RAID, if it can makes use of disk-spindle synchronization and its knowledge of the disk-platter position with regard to the disk head, and the desired disk-block. However, most modern (low-cost) disk drives do not offer this information and level of control anyway, and thus, most RAID hardware does not take advantage of it. RAID hardware is usually not compatible across different brands, makes and models: if a RAID controller fails, it must be replaced by another controller of the same type. As of this writing (June 1998), a broad variety of hardware controllers will operate under Linux; however, none of them currently come with configuration and management utilities that run under Linux.

소프트웨어 RAID는 커널 모듈로 설정하며, 관리 도구등도 모두 순수한 소프트웨어 적으로 이루어져 있다. 리눅스 RAID 시스템은 IDE, SCSI and Paraport drives 같은 저수준 드라이버와 block-device interface 위에 얇은 층으로 존재한다. ext2fs 나, DOS-FAT등의 파일시스템은 block-device interface위에 얻혀 있다. 소프트웨어 RAID는 소프트웨어적으로 매운 자연스러운 것이며, 하드웨어적 구현보다 유연한 것이다. 단점으로는 하드웨어 시스템보다 CPU cycle과 전원을 조금 더 소모한다는 것이지만, 가격이 비싸지는 것은 아니다. 소프트웨어 RAID는 파티션 단위로 움직이며, 각각의 파티션을 묶어서 RAID 파티션을 만들 수도 있다. 이것은 하드웨어적 구현과 크게 다른 점이며, 디스크들 전체를 하나로 묶어버릴 수도 있다. 그것은 하드웨어적으로는 운영체제로의 설정을 간단하고 명백히 할수 있고, 소프트웨어적으로는 좀 더 다양한 설정으로 복잡한 문제들에 접근할 수 있다.

Software-RAID is a set of kernel modules, together with management utilities that implement RAID purely in software, and require no extraordinary hardware. The Linux RAID subsystem is implemented as a layer in the kernel that sits above the low-level disk drivers (for IDE, SCSI and Paraport drives), and the block-device interface. The filesystem, be it ext2fs, DOS-FAT, or other, sits above the block-device interface. Software-RAID, by its very software nature, tends to be more flexible than a hardware solution. The downside is that it of course requires more CPU cycles and power to run well than a comparable hardware system. Of course, the cost can't be beat. Software RAID has one further important distinguishing feature: it operates on a partition-by-partition basis, where a number of individual disk partitions are ganged together to create a RAID partition. This is in contrast to most hardware RAID solutions, which gang together entire disk drives into an array. With hardware, the fact that there is a RAID array is transparent to the operating system, which tends to simplify management. With software, there are far more configuration options and choices, tending to complicate matters.

이 글이 쓰여지는 시점(1998년 6월)에서, Linux하의 RAID의 설정은 어려운 것이고, 숙련된 시스템 관리자가 설정하는 것이 좋을 것이다. 방법은 너무 복잡하고 , startup script들의 수정을 필요로 한다. 디스크 에러로부터의 복구는 평범한 것이 아니고 사람의 실수로 이어지기 쉽다. RAID는 초보자를 위한 것이 아니다. 속도 향상과 안전성을 얻기 전에 의외의 복잡함에 허를 찔리기 쉬우니 조심하기 바란다.. 특히, 요즘 디스크들은 믿을 수 없을 만큼 안전하고 요즘 CPU와 컨트롤러는 충분히 강력하다. 당신은 질좋고 빠른 하드웨어의 구입으로 좀더 쉽게 원하는 만큼의 속도와 안정성을 얻을 수 있을 것이다. As of this writing (June 1998), the administration of RAID under Linux is far from trivial, and is best attempted by experienced system administrators. The theory of operation is complex. The system tools require modification to startup scripts. And recovery from disk failure is non-trivial, and prone to human error. RAID is not for the novice, and any benefits it may bring to reliability and performance can be easily outweighed by the extra complexity. Indeed, modern disk drives are incredibly reliable and modern CPU's and controllers are quite powerful. You might more easily obtain the desired reliability and performance levels by purchasing higher-quality and/or faster hardware.

2. **Q**: RAID 레벨이 무엇인가요? 왜 그렇게 많은가요? 어떻게 구분하죠?

   A: 각 레벨마다, 속도와 사용공간, 안정성, 가격의 특성이 다르다. 모든 RAID 레벨의 과잉 사용공간이 디스크 오류를 대비해 주는 것은 아니다. RAID-1과 RAID-5가 가장 많이 사용되

며, RAID-1는 좀더 나은 속도를 내 줄 것이며, RAID-5는 좀 더 디스크의 여유공간을 많이 남겨줄것이다. 하지만, 속도가 레벨에 의해서 완전히 결정되는 것은 아니다. 속도는 사용할 프로그램, stripe,block,file 들의 크기등 다양한 요인에 많은 영향을 받는다. 이에 관해서는 이 뒤에서 자세히 다룰 것이다.

The different RAID levels have different performance, redundancy, storage capacity, reliability and cost characteristics. Most, but not all levels of RAID offer redundancy against disk failure. Of those that offer redundancy, RAID-1 and RAID-5 are the most popular. RAID-1 offers better performance, while RAID-5 provides for more efficient use of the available storage space. However, tuning for performance is an entirely different matter, as performance depends strongly on a large variety of factors, from the type of application, to the sizes of stripes, blocks, and files. The more difficult aspects of performance tuning are deferred to a later section of this HOWTO.

아래에서는 Linux 소프트웨어 RAID 구현의 다른 레벨들에 대해서 설명하고 있다.

The following describes the different RAID levels in the context of the Linux software RAID implementation.

- **선형 RAID (RAID-linear)** 은 여러개의 파티션들을 연결해 하나의 큰 가상 파티션을 만드는 것이다. 이것은 작은 드라이브들을 여러개 가지고 있고 이것을 하나의 큰 파티션으로 만들고자 할때 유용할 것이다. 하지만, 이 연결은 안전성을 제공하지 않는다. 하나의 디스크에 오류가 나면, 묶여있는 파티션 전체가 오류가 날것이다.
  is a simple concatenation of partitions to create a larger virtual partition. It is handy if you have a number small drives, and wish to create a single, large partition. This concatenation offers no redundancy, and in fact decreases the overall reliability: if any one disk fails, the combined partition will fail.
- **RAID-1** 는 "mirroring" 시키는 것이다. 두개 이상의 같은 크기를 가진 파티션들이 모두 블럭대 블럭으로 같은 데이터를 가지게 된다. 미러링은 디스크 오류에 아주 강력하다. 디스크 하나가 오류났을 때에도, 파손된 디스크와 완전히 똑같은 복제본이 있는 것이다. 미러링은 읽기 요청을 몇개의 디스크가 나누어 처리함으로써, I/O가 많은 시스템의 부하를 줄여줄수 있다. 하지만, 사용공간의 이용율에서 볼 때 미러링은 최악이다...
  is also referred to as "mirroring". Two (or more) partitions, all of the same size, each store an exact copy of all data, disk-block by disk-block. Mirroring gives strong protection against disk failure: if one disk fails, there is another with the an exact copy of the same data. Mirroring can also help improve performance in I/O-laden systems, as read requests can be divided up between several disks. Unfortunately, mirroring is also the least efficient in terms of storage: two mirrored partitions can store no more data than a single partition.
- **Striping** 은 다른 RAID 레벨에 기본적인 개념이다. stripe는 디스크 블럭들이 연속적으로 붙어있는 것이다. stripe 는 하나의 디스크 블럭만큼 짧을 수도 있을 것이고, 수 천개의 블럭들로 이루어져 있을 수도 있을 것이다. RAID 드라이버는 디스크 파티션을 stripe 로 나눌 것이다. RAID 의 레벨은 stripe가 어떻게 구성되었는가. 어떤 데이터를 담고 있는가에 따라서 달라질 것이다. stripe의 크기와, 파일시스템안의 파일의 크기, 그것들의 디스크안에서의 위치가 RAID 시스템의 전체적인 성능을 좌우할 것이다. (역자 덧, stripe는 띠인데.. 하나에 디스크에 있는게 아니라. 여러개의 디스크에서 같은 부분이 띠를 만드는 것이겠죠..)
  is the underlying concept behind all of the other RAID levels. A stripe is a contiguous sequence of disk blocks. A stripe may be as short as a single disk block, or may consist of thousands. The RAID drivers split up their component disk partitions into stripes; the different RAID levels differ in how they organize the stripes, and what data they put in them. The interplay between the size of the stripes, the typical size of files in the file system, and their location on the disk is what determines the overall performance of the RAID subsystem.

- **RAID-0** 은 선형 RAID에 더 가깝다. 파티션을 stripe 들로 나누고 묶는 것이다. 선형 RAID처럼 결과는 하나의 큰 파티션이고, 그것은 과잉 공간이 없다. 역시 안전성도 줄어든다. 단순한 선형 RAID에 비해 성능이 향상되긴 하지만, 파일 시스템과, stripe 의 크기에 의해 생기는 파일의 일반적인 크기, 작업의 형태에 많은 의존을 한다.
is much like RAID-linear, except that the component partitions are divided into stripes and then interleaved. Like RAID-linear, the result is a single larger virtual partition. Also like RAID-linear, it offers no redundancy, and therefore decreases overall reliability: a single disk failure will knock out the whole thing. RAID-0 is often claimed to improve performance over the simpler RAID-linear. However, this may or may not be true, depending on the characteristics to the file system, the typical size of the file as compared to the size of the stripe, and the type of workload. The `ext2fs` file system already scatters files throughout a partition, in an effort to minimize fragmentation. Thus, at the simplest level, any given access may go to one of several disks, and thus, the interleaving of stripes across multiple disks offers no apparent additional advantage. However, there are performance differences, and they are data, workload, and stripe-size dependent.
- **RAID-4** 는 RAID-0 처럼 stripe로 나누는 방식을 사용한다. 하지만, parity 정보를 저장할 추가적인 파티션을 사용한다. parity 는 과잉 정보를 저장하는데 사용되고, 하나의 디스크에 오류가 났을 때, 남은 디스크의 데이터는 파손된 디스크의 데이터를 복구하는데 사용될 것이다. N 개의 디스크가 있고, 하나의 parity 디스크가 있다면, parity stripe는 각 디스크의 stripe 들의 XOR 연산으로 계산될 것이다. (N+1) 디스크를 가진 RAID-4 배열의 저장용량은 N 이 될것이다. 하지만, RAID-4는 미러링만큼 읽는 속도가 빠르지 않고 매번 디스크를 쓸 때마다 연산을 하고 parity 디스크에 써야 한다. 때문에 쓰기가 많은 시스템에는 매번 parity 디스크를 access 해야 하기 때문에, 병목현상이 일어날 수 있다.
interleaves stripes like RAID-0, but it requires an additional partition to store parity information. The parity is used to offer redundancy: if any one of the disks fail, the data on the remaining disks can be used to reconstruct the data that was on the failed disk. Given N data disks, and one parity disk, the parity stripe is computed by taking one stripe from each of the data disks, and XOR'ing them together. Thus, the storage capacity of a an (N+1)-disk RAID-4 array is N, which is a lot better than mirroring (N+1) drives, and is almost as good as a RAID-0 setup for large N. Note that for N=1, where there is one data drive, and one parity drive, RAID-4 is a lot like mirroring, in that each of the two disks is a copy of each other. However, RAID-4 does **NOT** offer the read-performance of mirroring, and offers considerably degraded write performance. In brief, this is because updating the parity requires a read of the old parity, before the new parity can be calculated and written out. In an environment with lots of writes, the parity disk can become a bottleneck, as each write must access the parity disk.
- **RAID-5** 는 각 드라이브마다 parity stripe 를 저장시킴으로써 RAID-4의 쓰기 병목현상을 피할수 있다. 그리나, 여전히 쓰기 전에 XOR 연산을 해야 하기 때문에 쓰기 성능은 미러링만큼 빨라질수 없다. 읽기 역시 여러개의 데이터가 있는 것이 아니기 때문에 미러링만큼 빨라질 수 없다.
avoids the write-bottleneck of RAID-4 by alternately storing the parity stripe on each of the drives. However, write performance is still not as good as for mirroring, as the parity stripe must still be read and XOR'ed before it is written. Read performance is also not as good as it is for mirroring, as, after all, there is only one copy of the data, not two or more. RAID-5's principle advantage over mirroring is that it offers redundancy and protection against single-drive failure, while offering far more storage capacity when used with three or more drives.
- **RAID-2 와 RAID-3** 는 이제 거의 사용되지 않는다. 몇몇 레벨은 현대 디스크 기술로 인해 필요 없어졌기 때문이다. RAID-2는 RAID-4와 비슷하지만, parity 대신에 ECC 정보를 저장하는 것이 다르다. 현재의 모든 디스크들은 ECC 정보를 디스크 자체내에 넣어버렸다.

이것은, 디스크 자체에 작은 안전장치를 단 것이다. RAID-2 는 디스크 쓰기 도중 전원공급이 차단될 때, 데이터 안전성을 제공해준다. 하지만, 배터리 백업이나, clean shutdown 역시 똑같은 기능을 제공한다.. RAID-3은 가능한 최소의 stripe 크기를 사용하는 것을 제외하면 RAID-4 와 비슷하다. Linux 소프트웨어 RAID 드라이버는 RAID-2 와 RAID-3를 모두 지원하지 않는다.
are seldom used anymore, and to some degree are have been made obsolete by modern disk technology. RAID-2 is similar to RAID-4, but stores ECC information instead of parity. Since all modern disk drives incorporate ECC under the covers, this offers little additional protection. RAID-2 can offer greater data consistency if power is lost during a write; however, battery backup and a clean shutdown can offer the same benefits. RAID-3 is similar to RAID-4, except that it uses the smallest possible stripe size. As a result, any given read will involve all disks, making overlapping I/O requests difficult/impossible. In order to avoid delay due to rotational latency, RAID-3 requires that all disk drive spindles be synchronized. Most modern disk drives lack spindle-synchronization ability, or, if capable of it, lack the needed connectors, cables, and manufacturer documentation. Neither RAID-2 nor RAID-3 are supported by the Linux Software-RAID drivers.

- **그외의 RAID 레벨들은** 다양한 수요와 판매자들에 의해 만들어졌고, 특별한 하드웨어를 필요로 하거나 어떤 것들은 저작권을 보호 받고 있다. Linux 소프트웨어 RAID는 다른 어떤 변종들도 지원하지 않는다. have been defined by various researchers and vendors. Many of these represent the layering of one type of raid on top of another. Some require special hardware, and others are protected by patent. There is no commonly accepted naming scheme for these other levels. Sometime the advantages of these other systems are minor, or at least not apparent until the system is highly stressed. Except for the layering of RAID-1 over RAID-0/linear, Linux Software RAID does not support any of the other variations.

3 설치와 설정

1. **Q**: Software RAID 를 어떻게 설치해야 가장 좋을 까요?

   A: 나는 파일 시스템 계획이 좀 더 어려운 유닉스 설정작업인 것을 깨닫 도록 남겨둔다. 질문에 대한 대답으로, 우리가 한 일을 설명하겠다. 우리는 각각 2.1 기가의 EIDE 디스크를 아래와 같이 설정할 계획을 세웠다. I keep rediscovering that file-system planning is one of the more difficult Unix configuration tasks. To answer your question, I can describe what we did.

   We planned the following setup:

   - two EIDE disks, 2.1.gig each.

```
disk partition mount pt.  size    device
  1       1        /        300M   /dev/hda1
  1       2        swap      64M   /dev/hda2
  1       3        /home    800M   /dev/hda3
  1       4        /var     900M   /dev/hda4

  2       1        /root    300M   /dev/hdc1
  2       2        swap      64M   /dev/hdc2
  2       3        /home    800M   /dev/hdc3
  2       4        /var     900M   /dev/hdc4
```

   - 각 디스크는 모두 분리된 컨트롤러와 리본 케이블 상에 있다. 이것은 하나의 컨트롤러나 케이블이 고장 났을 때, 디스크들이 같이 사용 불가능하게 되는 것을 막아준다.

Each disk is on a separate controller (& ribbon cable). The theory is that a controller failure and/or ribbon failure won't disable both disks. Also, we might possibly get a performance boost from parallel operations over two controllers/cables.

- 루트 파티션 (/ /dev/hda1)에 리눅스 커널을 설치할 것이다. 이 파티션을 bootable로 설정해라.
  Install the Linux kernel on the root (/) partition /dev/hda1. Mark this partition as bootable.
- /dev/hac1은 /dev/hda1 의 RAID 복사본이 아닌 단순 복사본이다. 이것으로, 첫번째 디스크가 오류났을 때 rescue 디스크를 사용해 이 파티션을 bootable 설정하여 시스템을 다시 인스톨하지 않고 사용할 수 있다.
  /dev/hdc1 will contain a "cold" copy of /dev/hda1. This is NOT a raid copy, just a plain old copy-copy. It's there just in case the first disk fails; we can use a rescue disk, mark /dev/hdc1 as bootable, and use that to keep going without having to reinstall the system. You may even want to put /dev/hdc1's copy of the kernel into LILO to simplify booting in case of failure.
  이것은 심각한 문제 발생시, raid superblock-corruption 이나 다른 이해할수 없는 문제에 대한 걱정없이 시스템을 부팅할 수 있게 해준다.
  The theory here is that in case of severe failure, I can still boot the system without worrying about raid superblock-corruption or other raid failure modes & gotchas that I don't understand.
- /dev/hda3 와 /dev/hdc3 는 미러링을 통해 /dev/md0 가 될것이다. /dev/hda3 and /dev/hdc3 will be mirrors /dev/md0.
- /dev/hda4 와 /dev/hdc4 는 미러링을 통해 /dev/md1 가 될것이다. /dev/hda4 and /dev/hdc4 will be mirrors /dev/md1.
- 우리는 아래와 같은 이유로 파티션을 나누고, /var 와 /home 파티션을 미러링하기로 결정하였다. we picked /var and /home to be mirrored, and in separate partitions, using the following logic:
  - / (루트 파티션)의 데이터들은 상대적으로 잘 변하지 않는다.
    / (the root partition) will contain relatively static, non-changing data: for all practical purposes, it will be read-only without actually being marked & mounted read-only.
  - /home 파티션은 "천천히" 변하는 데이터를 가지고 있다. /home will contain "slowly" changing data.
  - /var> 는 메일 spool , 데이터베이스 내용, 웹 서버의 log 와 같은 급속히 변하는 데이터를 포함하고 있다. /var will contain rapidly changing data, including mail spools, database contents and web server logs.

  이렇게 여러개의 다른 파티션을 나누는 것은, 인간의 실수, 전원, 혹은 os의 문제등이 일어났을 때, 그것이 미치는 영향이 하나의 파티션에만 한정되기 때문이다.
  The idea behind using multiple, distinct partitions is that **if**, for some bizarre reason, whether it is human error, power loss, or an operating system gone wild, corruption is limited to one partition. In one typical case, power is lost while the system is writing to disk. This will almost certainly lead to a corrupted filesystem, which will be repaired by fsck during the next boot. Although fsck does it's best to make the repairs without creating additional damage during those repairs, it can be comforting to know that any such damage has been limited to one partition. In another typical case, the sysadmin makes a mistake during rescue operations, leading to erased or destroyed data. Partitions can help limit the repercussions of the operator's errors.
- /usr 와 /opt 파티션을 선택하여도 괜찮았을 것이다. 사실, 하드가 좀더 있었다면, /opt 와 /home 파티션을 RAID-5 로 설정하는 것이 더 좋았을 것이다. 주의할 점은 /usr 파티션을

RAID-5로 설정하지 말라는 것이다. 심각한 문제가 일어났을 경우 `/usr` 파티션에 마운트 할수 없게 될 것이고, `/usr` 파티션안의 네트워크 툴과 컴파일러 같은 것들을 필요로 하게 될 것이다. RAID-1을 사용한다면, 이런 에러가 났을때, RAID는 사용할수 없어도 두개의 미러링된 것중 하나에는 마운트가 가능하다. Other reasonable choices for partitions might be `/usr` or `/opt`. In fact, `/opt` and `/home` make great choices for RAID-5 partitions, if we had more disks. A word of caution: **DO NOT** put `/usr` in a RAID-5 partition. If a serious fault occurs, you may find that you cannot mount `/usr`, and that you want some of the tools on it (e.g. the networking tools, or the compiler.) With RAID-1, if a fault has occurred, and you can't get RAID to work, you can at least mount one of the two mirrors. You can't do this with any of the other RAID levels (RAID-5, striping, or linear append).

그래서 질문에 대한 완성된 답은:

- 첫번째 디스크의 첫번째 파티션에 운영체제를 설치하고 다른 파티션들은 마운트하지 말아라.install the OS on disk 1, partition 1. do NOT mount any of the other partitions.
- 명령단위로 RAID를 설치하라.install RAID per instructions.
- `md0` 와 `md1`. 설정하라.configure `md0` and `md1`.
- 디스크 오류가 일어났을 때 무엇을 해야 하는 지 준비해라. 관리자가 지금 실수하는지 찾아보고, 타격을 입게 놔두지 마라. 그리고 경험을 쌓아라. (우리는 디스크가 작동하고 있는 동안, 전원을 꺼보았다. 이것은 멍청해보이지만, 정보를 얻을 수 있다.) convince yourself that you know what to do in case of a disk failure! Discover sysadmin mistakes now, and not during an actual crisis. Experiment! (we turned off power during disk activity — this proved to be ugly but informative).
- `/var` 를 `/dev/md1`으로 옮기는 중, 어느 정도 잘못된 mount/copy/unmount/rename/reboot 을 해보라. 조심히만 한다면, 위험하지는 않을 것이다.do some ugly mount/copy/unmount/rename/reboot scheme to move `/var` over to the `/dev/md1`. Done carefully, this is not dangerous.
- 그리고, 그것들을 즐겨라.

2. **Q**: `mdadd`, `mdrun` 등의 명령과 `raidadd`, `raidrun` 명령의 다른 점이 뭔가요?

   A: raidtools 패키지의 0.5 버젼부터 이름이 바뀌었다. `md`로 이름이 붙는 것은 0.43 이전버젼이고 `raid`로 이름이 붙는 것은 0.5 버젼과 더 새버젼들이다.. The names of the tools have changed as of the 0.5 release of the raidtools package. The `md` naming convention was used in the 0.43 and older versions, while `raid` is used in 0.5 and newer versions.

3. **Q**: 가지고 있는 2.0.34 커널에서 RAID-linear 와 RAID-0 를 사용하고 싶다. RAID-linear 와 RAID-0 을 위해서 패치가 필요하지 않기 때문에. raid 패치는 하고 싶지 않다. 어디에 가면, 이것들을 위한 raid-tool 을 구할수 있나?

   A: 과격한 질문이다. 사실, 최신의 raid tool들은 컴파일 하기 위해 RAID-1,4,5 커널 패치를 필요로 한다. 현재 raid tool의 컴파일된 바이너리 버젼찾지 못했다. 하지만, 2.1.100 커널에서 컴파일된 바이너리가 2.0.34 커널에서 RAID-0/linear 파티션을 만드는 것을 잘 수행하는 것을 보았다. 그래서, 나는 `http://linas.org/linux/Software-RAID/` 에 mdadd,mdcreate등의 바이너리를 임시적으로 올린다. This is a tough question, indeed, as the newest raid tools package needs to have the RAID-1,4,5 kernel patches installed in order to compile. I am not aware of any pre-compiled, binary version of the raid tools that is available at this time. However, experiments show that the raid-tools binaries, when compiled against kernel 2.1.100, seem to work just fine in creating a RAID-0/linear partition under 2.0.34. A brave soul has asked for these, and I've **temporarily** placed the binaries mdadd, mdcreate, etc. at http://linas.org/linux/Software-RAID/ You must get the man pages, etc. from the usual raid-tools package.

4. **Q**: 루트 파티션에 RAID를 적용할 수 있는가? 왜 md 디스크로 직접 부팅할 수 없는가?

A:

LILO와 Loadlin 모두 RAID 파티션에서 커널이미지를 읽어올 수 없다. 루트 파티션에 RAID를 적용하고 싶다면, 커널을 저장할 RAID가 아닌 파티션을 만들어야 할것이다. (일반적으로 이 파티션의 이름은 `/boot`이다.) <`HarryH@Royal.Net`> 로부터 받은 initial ramdisk (initrd) 또는, 패치는 RAID 디스크를 root 디바이스로 사용가능하게 해 줄것이다. (이 패치는 최근 2.1.x커널에는 기본적으로 채택되어있다.)

Both LILO and Loadlin need an non-stripped/mirrored partition to read the kernel image from. If you want to strip/mirror the root partition (/), then you'll want to create an unstriped/mirrored partition to hold the kernel(s). Typically, this partition is named `/boot`. Then you either use the initial ramdisk support (initrd), or patches from Harald Hoyer <`HarryH@Royal.Net`> that allow a stripped partition to be used as the root device. (These patches are now a standard part of recent 2.1.x kernels)

거기에는 사용할 수 있는 몇가지 방법이 있는데, 하나는 Bootable RAID mini-HOWTO: `<ftp://ftp.bizsystems.com/pub/raid/bootable-raid>`에 자세히 설명되어 있다.

There are several approaches that can be used. One approach is documented in detail in the Bootable RAID mini-HOWTO: `<ftp://ftp.bizsystems.com/pub/raid/bootable-raid>`.

또는, 아래처럼 `mkinitrd` 를 사용해 ramdisk image를 만들수도 있다.

Alternately, use `mkinitrd` to build the ramdisk image, see below.

Edward Welbon <`welbon@bga.com`> writes:

- ... all that is needed is a script to manage the boot setup. To mount an `md` filesystem as root, the main thing is to build an initial file system image that has the needed modules and md tools to start `md`. I have a simple script that does this.
- For boot media, I have a small **cheap** SCSI disk (170MB I got it used for $20). This disk runs on a AHA1452, but it could just as well be an inexpensive IDE disk on the native IDE. The disk need not be very fast since it is mainly for boot.
- This disk has a small file system which contains the kernel and the file system image for `initrd`. The initial file system image has just enough stuff to allow me to load the raid SCSI device driver module and start the raid partition that will become root. I then do an

  ```
  echo 0x900 > /proc/sys/kernel/real-root-dev
  ```

  (`0x900` is for `/dev/md0`) and exit `linuxrc`. The boot proceeds normally from there.
- I have built most support as a module except for the AHA1452 driver that brings in the `initrd` filesystem. So I have a fairly small kernel. The method is perfectly reliable, I have been doing this since before 2.1.26 and have never had a problem that I could not easily recover from. The file systems even survived several 2.1.4[45] hard crashes with no real problems.
- At one time I had partitioned the raid disks so that the initial cylinders of the first raid disk held the kernel and the initial cylinders of the second raid disk hold the initial file system image, instead I made the initial cylinders of the raid disks swap since they are the fastest cylinders (why waste them on boot?).
- The nice thing about having an inexpensive device dedicated to boot is that it is easy to boot from and can also serve as a rescue disk if necessary. If you are interested, you can take a look at the script that builds my initial ram disk image and then runs `LILO`.

  `<http://www.realtime.net/welbon/initrd.md.tar.gz>`

  It is current enough to show the picture. It isn't especially pretty and it could certainly build a much smaller filesystem image for the initial ram disk. It would be easy to a make it more efficient. But it uses LILO as is. If you make any improvements, please forward a copy to me. 8-)

5. **Q**: striping 위에 미러링이 가능하다고 들었는데, 사실인가? loopback 장치로 미러링할 수 있는가?

   A: 그렇다. 하지만, 그 반대로는 안된다. Yes, but not the reverse. That is, you can put a stripe over several disks, and then build a mirror on top of this. However, striping cannot be put on top of mirroring.

   간단히 기술적인 설명을 덧붙이자면, linear 와 stripe는 자체적으로 ll_rw_blk 루틴을 사용하는 데 이것은 block 를 사용하지 않고 디스크 device와 sector를 사용해 정식적으로, 그리고 저수준의 access를 한다, 때문에, 다른 미러링위에 위치시킬수 없다.

   A brief technical explanation is that the linear and stripe personalities use the ll_rw_blk routine for access. The ll_rw_blk routine maps disk devices and sectors, not blocks. Block devices can be layered one on top of the other; but devices that do raw, low-level disk accesses, such as ll_rw_blk, cannot.

   현재 (1997년 11월) RAID는 loopback device를 지원하지 않지만, 곧 지원할 것이다.

   Currently (November 1997) RAID cannot be run over the loopback devices, although this should be fixed shortly.

6. **Q**: 두개의 작은 디스크와 세개의 큰 디스크를 가지고 있을때, 작은 디스크 두개를 RAID-0으로 묶은 후, 나머지 디스크들과, RAID-5를 만들수 있는가?

   A: 1997년 11월 현재, RAID-5로 묶을 수는 없다. 묶여진 디스크들로는 RAID-1(미러링)만 가능하다. Currently (November 1997), for a RAID-5 array, no. Currently, one can do this only for a RAID-1 on top of the concatenated drives.

7. **Q**: 두개의 디스크로 RAID-1 을 설정하는 것과, RAID-5를 설정하는 것이 어떻게 다른가?

   A: 데이터의 저장율에는 차이가 없다. 디스크를 더 붙힌다고 저장율이 늘어가는 것도 아니다. There is no difference in storage capacity. Nor can disks be added to either array to increase capacity (see the question below for details).

   RAID-1 은 각 드라이브에서 두 섹터를 동시에 읽는 분산 기술을 사용하기 때문에 두배의 읽기 성능을 보여준다.

   RAID-1 offers a performance advantage for reads: the RAID-1 driver uses distributed-read technology to simultaneously read two sectors, one from each drive, thus doubling read performance.

   RAID-5는 많은 것들을 포함하지만, 1997년 9월 현재 까지는, 데이터 디스크가 parity 디스크로 실제적으로 미러링되지는 않는다. 때문에 데이터를 병렬로 읽지는 않는다.

   The RAID-5 driver, although it contains many optimizations, does not currently (September 1997) realize that the parity disk is actually a mirrored copy of the data disk. Thus, it serializes data reads.

8. **Q**:

   두개의 디스크가 망가졌을때에는 어떻게 대비하죠?

   A: 몇몇의 RAID 는 알고리즘은 여러개의 디스크가 망가졌을 때를 대비할 수 있다. 하지만, 현재 리눅스에서 지원되지는 않는다. 그러나, RAID위에 RAID를 구축함으로써, Linux Software RAID로도, 그런 상황에 대비할 수 있다. 예를 들면,9개의 디스크로 3개의 RAID-5를 만들고 다시 그것을 하나의 RAID-5 로 만드는 것이다. 이런 설정은 3개의 디스크가 망가졌을때까지 대비할 수 있지만, 많은 공간이 "낭비"된다는 것을 주목하라. Some of the RAID algorithms do guard against multiple disk failures, but these are not currently implemented for Linux. However, the Linux Software RAID can guard against multiple disk failures by layering an array on top of an array. For example, nine disks can be used to create three raid-5 arrays. Then these three arrays can in turn be hooked together into a single RAID-5 array on top. In fact, this kind of a configuration will guard against a three-disk failure. Note that a large amount of disk space is "wasted" on the redundancy information.

```
For an NxN raid-5 array,
N=3, 5 out of 9 disks are used for parity (=55%)
N=4, 7 out of 16 disks
N=5, 9 out of 25 disks
...
N=9, 17 out of 81 disks (=~20%)
```

일반적으로, MxN 개로 만들어진 RAID를 위해 M+N-1 개의 디스크가 parity 로 사용되고, M = N 일때 버려지는 양이 최소가 될 것이다.

In general, an MxN array will use M+N-1 disks for parity. The least amount of space is "wasted" when M=N.

다른 방법은 세개의 디스크(RAID-5로 설정된)로 RAID-1을 만드는 것이다. 그것은, 세개의 디스크중 같은 데이터를 가지는 2/3을 낭비하게 될 것이다.

Another alternative is to create a RAID-1 array with three disks. Note that since all three disks contain identical data, that 2/3's of the space is "wasted".

9. **Q**: 파티션이 제대로 unmount 되지 않았을 때 `fsck`가 실행되어서 파일시스템을 스스로 고치는 것이 어떻게 가능한지 알고 싶다. RAID 시스템을 `ckraid --fix` 로 고칠수 있는데 왜 그것을 자동으로 하지 않는가? I'd like to understand how it'd be possible to have something like `fsck`: if the partition hasn't been cleanly unmounted, `fsck` runs and fixes the filesystem by itself more than 90% of the time. Since the machine is capable of fixing it by itself with `ckraid --fix`, why not make it automatic?

   A: `/etc/rc.d/rc.sysinit` 에 아래와 같이 추가함으로써 할수 있다.

This can be done by adding lines like the following to `/etc/rc.d/rc.sysinit`:

```
mdadd /dev/md0 /dev/hda1 /dev/hdc1 || {
    ckraid --fix /etc/raid.usr.conf
    mdadd /dev/md0 /dev/hda1 /dev/hdc1
}
```

or

```
mdrun -p1 /dev/md0
if [ $? -gt 0 ] ; then
        ckraid --fix /etc/raid1.conf
        mdrun -p1 /dev/md0
fi
```

좀더 완벽한 스크립트를 만들기 이전에 시스템이 어떻게 켜지는지 보도록 하자.

Before presenting a more complete and reliable script, lets review the theory of operation.

정상적으로 종료되지 않았다면, 리눅스는 아래와 같은 상태중의 하나일 꺼라고 Gadi Oxman은 말했다.

Gadi Oxman writes: In an unclean shutdown, Linux might be in one of the following states:

- 비정상 종료당시, 메모리의 디스크 캐쉬가 저장(sync) 된 상태. 데이터는 손상되지 않는다. The in-memory disk cache was in sync with the RAID set when the unclean shutdown occurred; no data was lost.

- 문제가 발생했을 때, 디스크 캐쉬는 RAID 에 있는 것보다 최근 것이었던 상태 이 결과는 파일시스템이 망가지고, 데이터를 잃을 것이다. 이것은 다시 아래의 두가지 상태로 나빰墨厤 The in-memory disk cache was newer than the RAID set contents when the crash occurred; this results in a corrupted filesystem and potentially in data loss.

  This state can be further divided to the following two states:

  – 리눅스가 데이터를 쓰고(write) 있었을 경우.
  – 리눅스가 데이터를 쓰고 있지 않았을 경우.

RAID-1을 사용한다면, 위의 첫번째 경우에서, 어느정도만 미러링됐을 경우가 생긴다. 이런 경우, 다음 부팅때, 미러링된 데이터가 서로 같지 않을 것이다.

이런경우에 미러링이 다른걸 무시한다면, 읽기시 미러링된 것중 하나를 선택할 것이고, 모순된 결과를 출력할 것이다.

Suppose we were using a RAID-1 array. In (2a), it might happen that before the crash, a small number of data blocks were successfully written only to some of the mirrors, so that on the next reboot, the mirrors will no longer contain the same data.

If we were to ignore the mirror differences, the raidtools-0.36.3 read-balancing code might choose to read the above data blocks from any of the mirrors, which will result in inconsistent behavior (for example, the output of `e2fsck -n /dev/md0` can differ from run to run).

RAID 는 비정상적인 shutdown을 위해 설계된 것이 아니고, 일반적으로 미러링된 데이터가 다를 때나, 파일시스템이 고장났을 때의 완벽한 해결책도 없다.

Since RAID doesn't protect against unclean shutdowns, usually there isn't any "obviously correct" way to fix the mirror differences and the filesystem corruption.

For example, by default `ckraid --fix` will choose the first operational mirror and update the other mirrors with its contents. However, depending on the exact timing at the crash, the data on another mirror might be more recent, and we might want to use it as the source mirror instead, or perhaps use another method for recovery.

아래의 스크립트를 `rc.raid.init` 에 추가하고, 그 디렉토리에 path를 걸어라. 그것은 좀더 안전한 부팅을 지원할 것이고, 특히, 일치하지 않는 디스크나, 콘트롤러, 콘트롤러 드라이버등이 있으면, 질고 반복적으로 `chraid`를 실행할 것이다. `rc.raid.init`는 `fsck`로 루트파티션이 체크되고 Read Write 마운트 된 상태에서 작동할 것이다.

The following script provides one of the more robust boot-up sequences. In particular, it guards against long, repeated `ckraid`'s in the presence of uncooperative disks, controllers, or controller device drivers. Modify it to reflect your config, and copy it to `rc.raid.init`. Then invoke `rc.raid.init` after the root partition has been fsck'ed and mounted rw, but before the remaining partitions are fsck'ed. Make sure the current directory is in the search path.

```
mdadd /dev/md0 /dev/hda1 /dev/hdc1 || {
    rm -f /fastboot             # force an fsck to occur
    ckraid --fix /etc/raid.usr.conf
    mdadd /dev/md0 /dev/hda1 /dev/hdc1
}
# if a crash occurs later in the boot process,
# we at least want to leave this md in a clean state.
/sbin/mdstop /dev/md0

mdadd /dev/md1 /dev/hda2 /dev/hdc2 || {
    rm -f /fastboot             # force an fsck to occur
    ckraid --fix /etc/raid.home.conf
```

```
        mdadd /dev/md1 /dev/hda2 /dev/hdc2
    }
    # if a crash occurs later in the boot process,
    # we at least want to leave this md in a clean state.
    /sbin/mdstop /dev/md1

    mdadd /dev/md0 /dev/hda1 /dev/hdc1
    mdrun -p1 /dev/md0
    if [ $? -gt 0 ] ; then
        rm -f /fastboot             # force an fsck to occur
        ckraid --fix /etc/raid.usr.conf
        mdrun -p1 /dev/md0
    fi
    # if a crash occurs later in the boot process,
    # we at least want to leave this md in a clean state.
    /sbin/mdstop /dev/md0

    mdadd /dev/md1 /dev/hda2 /dev/hdc2
    mdrun -p1 /dev/md1
    if [ $? -gt 0 ] ; then
        rm -f /fastboot             # force an fsck to occur
        ckraid --fix /etc/raid.home.conf
        mdrun -p1 /dev/md1
    fi
    # if a crash occurs later in the boot process,
    # we at least want to leave this md in a clean state.
    /sbin/mdstop /dev/md1

    # OK, just blast through the md commands now.  If there were
    # errors, the above checks should have fixed things up.
    /sbin/mdadd /dev/md0 /dev/hda1 /dev/hdc1
    /sbin/mdrun -p1 /dev/md0

    /sbin/mdadd /dev/md12 /dev/hda2 /dev/hdc2
    /sbin/mdrun -p1 /dev/md1
```

아래와 같이 `rc.raid.halt`를 추가하고 싶다면 추가하라.

In addition to the above, you'll want to create a `rc.raid.halt` which should look like the following:

```
    /sbin/mdstop /dev/md0
    /sbin/mdstop /dev/md1
```

`rc.sysinit`와 `init.d/halt` 스크립트중 시스템의 halt/roboot를 위한 모든 unmount 전에 위 파일을 첨부시켜라. (`rc.sysinit`에는 fsck가 실패했을때, 디스크를 unmount 하고 reboot 하는 루틴이 있다.)

Be sure to modify both `rc.sysinit` and `init.d/halt` to include this everywhere that filesystems get unmounted before a halt/reboot. (Note that `rc.sysinit` unmounts and reboots if `fsck` returned with an error.)

10. **Q**: 현재 가진 하나의 디스크로 반쪽짜리 RAID-1을 구성후, 나중에 디스크를 추가할수 있나요?

    A: 현재의 도구들로는 불가능하고, 어떤 쉬운 방법도 없다. 특히, 디스크의 내용을 단지 복사함으로써, 미러링은 이루어지지 않는다. RAID 드라이버는 피티션끝의 작은 공간을 superblock로 사용하기 때문이다. 이것은 작은 공간만을 차지하지만, 이미 존재하는 파일 시스테에 단순히 복사하려 할경우, superblock은 파일 시스템을 덮어씌울 것이고, 데이터를 엉망으로 만들것이다. ext2fs 파일시스템은 파일들이 조각나는 것을 막기위해, 파일들을 무작위로 배치시켜왔기 때문에, 디스크를 모두 사용하기 전에, 파티션의 끝부분이 충분히 사용될 수 있다.

    With the current tools, no, not in any easy way. In particular, you cannot just copy the contents of one disk onto another, and then pair them up. This is because the RAID drivers use glob of space at the end of the partition to store the superblock. This decreases the amount of space available to the file system slightly; if you just naively try to force a RAID-1 arrangement onto a partition with an existing filesystem, the raid superblock will overwrite a portion of the file system and mangle data. Since the ext2fs filesystem scatters files randomly throughput the partition (in order to avoid fragmentation), there is a very good chance that some file will land at the very end of a partition long before the disk is full.

    당신이 유능하다면, superblock이 어느정도의 공간을 차지하는지 계산해서, 파일시스템을 조금 작게 만들것을 제안한다. 그리고, 디스크를 추가할때, RAID 툴을 당신에 맞게 고쳐서 사용해야 할것이다. (그 툴들이 끔찍하게 복잡하지는 않다.)

    If you are clever, I suppose you can calculate how much room the RAID superblock will need, and make your filesystem slightly smaller, leaving room for it when you add it later. But then, if you are this clever, you should also be able to modify the tools to do this automatically for you. (The tools are not terribly complex).

    주의깊게 읽은 사람이라면, 아래와 같은 것이 작동할것이라고 지적했을 것이다. 나는 이것을 시도해보거나 증명해보지는 못했다. `/dev/null`을 하나의 기기로써 `mkraid`에 이용하는 것이다. 진짜 디스크 하나를 가지고 `mdadd -r` 를 실행시킨후, `mkraid`로 RAID 배열을 만들수 있을 것이고, 디스크 하나가 깨졌을 때처럼 "degraded" 모드로 작동시킬수 있을것이다.

    Note:A careful reader has pointed out that the following trick may work; I have not tried or verified this: Do the `mkraid` with `/dev/null` as one of the devices. Then `mdadd -r` with only the single, true disk (do not mdadd `/dev/null`). The `mkraid` should have successfully built the raid array, while the mdadd step just forces the system to run in "degraded" mode, as if one of the disks had failed.

4 에러의 복구

1. **Q**:

   RAID-1을 사용하고 있는데, 디스크가 작동중 전원이 꺼졌습니다. 어떻게 해야 할까요?

   A: 이런 상황에서는 몇가지 방법이 있다. The redundancy of RAID levels is designed to protect against a **disk** failure, not against a **power** failure.

   There are several ways to recover from this situation.

   - 첫번째 방법은 raid 도구들을 사용하는 것이다. 이것은 raid의 데이터들을 동기화 시켜준다.(sync) 하지만, 파일시스템의 손상은 복구해주지 않으므로 후에 fsck를 사용해 고쳐야 한다. RAID 는 `ckraid /etc/raid1.conf` 통해 점검해볼수 있다.(RAID-1일 경우이다, 다른 경우라면, `/etc/raid5.conf`처럼사용해야 한다.) `ckraid /etc/raid1.conf --fix` 를 사용해 RAID된 디스크중 하나를 선택해서, 다른 디스크로 미러링 시킬수 있다. 디스크중 어느 것을 선택해야 할지 모른다면, `ckraid /etc/raid1.conf --fix --force-source /dev/hdc3` 이런 식으로 `--force-source`옵션을 사용하라. ckraid 는 `--fix` 옵션을 제거함으로써 RAID 시스템에 어떤 변화없이 안전하게 시도 될 수 있다. 제안된 변경에 대해서, 만족할 경우에 `--fix` 옵션을 사용해라.

Method (1): Use the raid tools. These can be used to sync the raid arrays. They do not fix file-system damage; after the raid arrays are sync'ed, then the file-system still has to be fixed with fsck. Raid arrays can be checked with `ckraid /etc/raid1.conf` (for RAID-1, else, `/etc/raid5.conf`, etc.)
Calling `ckraid /etc/raid1.conf --fix` will pick one of the disks in the array (usually the first), and use that as the master copy, and copy its blocks to the others in the mirror.
To designate which of the disks should be used as the master, you can use the `--force-source` flag: for example, `ckraid /etc/raid1.conf --fix --force-source /dev/hdc3`
The ckraid command can be safely run without the `--fix` option to verify the inactive RAID array without making any changes. When you are comfortable with the proposed changes, supply the `--fix` option.

- 두번째 방법은 첫번째 방법보다 많이 좋은 방법은 아니다. `/dev/hda3` 와 `/dev/hdc3` 로 만들어진 RAID-1 디스크가 있다고 가정할때, 아래와 같이 해볼수 있다. Method (2): Paranoid, time-consuming, not much better than the first way. Lets assume a two-disk RAID-1 array, consisting of partitions `/dev/hda3` and `/dev/hdc3`. You can try the following:
  (a) `fsck /dev/hda3`
  (b) `fsck /dev/hdc3`
  (c) 두개의 파티션중, 에러가 적은 쪽이나, 더 쉽게 복구가 된쪽, 또는 복구하고 싶은 데이터가 남아있는 쪽등, 새로운 마스터로 쓸 파티션을 결정해야 한다. `/dev/hdc3` 를 선택했다 하자. decide which of the two partitions had fewer errors, or were more easily recovered, or recovered the data that you wanted. Pick one, either one, to be your new "master" copy. Say you picked `/dev/hdc3`.
  (d) `dd if=/dev/hdc3 of=/dev/hda3`
  (e) `mkraid raid1.conf -f --only-superblock`

  마지막 두단계 대신에 `ckraid /etc/raid1.conf --fix --force-source /dev/hdc3` 를 사용하면 좀 더 빠를 것이다.
  Instead of the last two steps, you can instead run `ckraid /etc/raid1.conf --fix --force-source /dev/hdc3` which should be a bit faster.
- 세번㈁⑨ 방법은 오랫동안 fsck를 기다리기가 귀찮은 사람들을 위한 것이다. 첫번째 3단계를 뛰어넘고 바로 마지막 두단계를 실행하는 것이다. 그런 후에 `fsck /dev/md0` 를 실행하는 것이다. 이것은 첫번째 방법의 모양을 바꾼 것일 뿐이다. Method (3): Lazy man's version of above. If you don't want to wait for long fsck's to complete, it is perfectly fine to skip the first three steps above, and move directly to the last two steps. Just be sure to run `fsck /dev/md0` after you are done. Method (3) is actually just method (1) in disguise.

어떤 방법도 RAID를 동기화 시켜줄 수 있을 뿐이고, 파일 시스템 또한 잘 복구되기를 원할 것이다. 이를 위해서, fsck를 md device를 unmount 시킨후 fsck를 실행하라.

In any case, the above steps will only sync up the raid arrays. The file system probably needs fixing as well: for this, fsck needs to be run on the active, unmounted md device.

세개의 디스크로 구성된 RAID-1 시스템이라면 많이 일치한 부분을 통해, 답을 찾아내는 방법등의, 조금 더 많은 방법이 있겠지만, 이런 것을 자동으로 해주는 도구는 현재 지원되지 않는다.

With a three-disk RAID-1 array, there are more possibilities, such as using two disks to "vote" a majority answer. Tools to automate this do not currently (September 97) exist.

2. **Q**: RAID-4 또는 RAID-5 시스템을 가지고 있는데, 디스크 작동중에 꺼졌습니다. 어떻게 해야 할까요?

A: RAID-4나 RAID-5 시스템에서는 예비 수리를 위해 fsck를 사용할 수 없다. 먼저 ckraid를 사용하라. ckraid 는 `--fix` 옵션을 제거함으로써 RAID 시스템에 어떤 변화없이 안전하게 시도 될 수 있다. 제안된 변경에 대해서, 만족할 경우에 `--fix` 옵션을 사용해라.

원한다면,`--suggest-failed-disk-mask` 옵션을 통해 디스크들중 하나를 망가진 디스크로 지정한 채 ckraid를 시도 할수 있다.

RAID-5는 단지 하나의 bit만이 flag 로 설정되어 있기 때문에, RAID-5는 두개의 디스크가 망가졌을 때는 복구할 수 없다. 아래는 binary bit mask 이다.

The redundancy of RAID levels is designed to protect against a **disk** failure, not against a **power** failure.

Since the disks in a RAID-4 or RAID-5 array do not contain a file system that fsck can read, there are fewer repair options. You cannot use fsck to do preliminary checking and/or repair; you must use ckraid first.

The `ckraid` command can be safely run without the `--fix` option to verify the inactive RAID array without making any changes. When you are comfortable with the proposed changes, supply the `--fix` option.

If you wish, you can try designating one of the disks as a "failed disk". Do this with the `--suggest-failed-disk-mask` flag.

Only one bit should be set in the flag: RAID-5 cannot recover two failed disks. The mask is a binary bit mask: thus:

```
0x1 == first disk
0x2 == second disk
0x4 == third disk
0x8 == fourth disk, etc.
```

또는, `--suggest-fix-parity` 옵션을 통해 parity 섹터를 수정할 수도 있다. 이것은 다른 섹터들로부터 parity 를 다시 계산해낼 것이다.

`--suggest-failed-dsk-mask` 와 `--suggest-fix-parity` 옵션은 `--fix`옵션을 제거함으로써, 가능한 수정 계획의 확인을 위해 안전하게 사용될수 있다.

Alternately, you can choose to modify the parity sectors, by using the `--suggest-fix-parity` flag. This will recompute the parity from the other sectors.

The flags `--suggest-failed-dsk-mask` and `--suggest-fix-parity` can be safely used for verification. No changes are made if the `--fix` flag is not specified. Thus, you can experiment with different possible repair schemes.

3. **Q**: `/dev/hda3` 과 `/dev/hdc3` 두개의 디스크로 `/dev/md0` 의 RAID-1 시스템을 만들어서 사용하고 있습니다. 최근에, `/dev/hdc3` 이 고장나서 새 디스크를 구입했다. 제 가장 친한 친구가, "`dd if=/dev/hda3 of=/dev/hdc3`"를 해보라고 해서 해보았지만, 아직도 작동하고 있지 않습니다. My RAID-1 device, `/dev/md0` consists of two hard drive partitions: `/dev/hda3` and `/dev/hdc3`. Recently, the disk with `/dev/hdc3` failed, and was replaced with a new disk. My best friend, who doesn't understand RAID, said that the correct thing to do now is to "`dd if=/dev/hda3 of=/dev/hdc3`". I tried this, but things still don't work.

   A: 친구를 당신의 컴퓨터에 가까이 가게 하지 않게 해서, 교우관계를 유지하는 게 좋을 것이다. 다행스럽게도, 심각한 손상을 입지는 않는다. 아래와 같이 실행함으로써, 시스템을 회복시킬수 있을 것이다.

   ```
   mkraid raid1.conf -f --only-superblock
   ```

   dd 명령어를 이용해서, 파티션의 복사본을 만드는 것은 대부분 가능하다. 하지만, RAID-1 시스템에서는 superblock 이 다르기 때문에 안된다. 때문에 RAID-1을 두 파티션중의 하나의 superblock를 다시 만들어주면, 다시 사용가능하게 될 것이다. You should keep your best friend away from you computer. Fortunately, no serious damage has been done. You can recover from this by running:

```
mkraid raid1.conf -f --only-superblock
```

By using dd, two identical copies of the partition were created. This is almost correct, except that the RAID-1 kernel extension expects the RAID superblocks to be different. Thus, when you try to reactivate RAID, the software will notice the problem, and deactivate one of the two partitions. By re-creating the superblock, you should have a fully usable system.

4. **Q**: 내 mkraid 는 --only-superblock 옵션이 지원되지 않는 버젼입니다. 어떻게 해야 할까요?

   A: 새로운 툴에서는 --force-resync 옵션으로 바뀌었고, 최신의 툴들의 사용은 아래와 같이 사용해야 한다.

   ```
   umount /web (/dev/md0가 마운트 되어있는 곳.)
   raidstop /dev/md0
   mkraid /dev/md0 --force-resync --really-force
   raidstart /dev/md0
   ```

   cat /proc/mdstat 를 통해 결과를 볼 수 있을 것이고, mount /dev/md0를 통해 다시 사용가능할 것이다.

   The newer tools drop support for this flag, replacing it with the --force-resync flag. It has been reported that the following sequence appears to work with the latest tools and software:

   ```
   umount /web (where /dev/md0 was mounted on)
   raidstop /dev/md0
   mkraid /dev/md0 --force-resync --really-force
   raidstart /dev/md0
   ```

   After doing this, a cat /proc/mdstat should report resync in progress, and one should be able to mount /dev/md0 at this point.

5. **Q**: /dev/hda3 과 /dev/hdc3 두개의 디스크로 /dev/md0 의 RAID-1 시스템을 만들어서 사용하고 있습니다. 제 가장 친한 (여자?) 친구가, 못보는 사이, /dev/hda3 를 fsck 로 실행시키는 바람에, RAID가 동작하지 않고 있습니다. 어떻게 해야 할까요? My RAID-1 device, /dev/md0 consists of two hard drive partitions: /dev/hda3 and /dev/hdc3. My best (girl?)friend, who doesn't understand RAID, ran fsck on /dev/hda3 while I wasn't looking, and now the RAID won't work. What should I do?

   A: 당신은 가장 친한 친구라는 개념을 다시 한번 생각해보아야 할것이다. 일반적으로 fsck 는 RAID를 만드는 파티션중 하나에서 돌려서는 절대로 안된다. 파티션 손상이나, 데이터 손상이 발생되지 않았다고 한다면, RAID-1 시스템을 아래와 같이 수리할 수 있다.

   (a) /dev/hda3 의 백업을 받는다.

   (b) dd if=/dev/hda3 of=/dev/hdc3

   (c) mkraid raid1.conf -f --only-superblock

   You should re-examine your concept of "best friend". In general, fsck should never be run on the individual partitions that compose a RAID array. Assuming that neither of the partitions are/were heavily damaged, no data loss has occurred, and the RAID-1 device can be recovered as follows:

   (a) make a backup of the file system on /dev/hda3

   (b) dd if=/dev/hda3 of=/dev/hdc3

   (c) mkraid raid1.conf -f --only-superblock

   This should leave you with a working disk mirror.

6. **Q**:

   왜 위의 복구 순서대로 해야 하는가?

   A: RAID-1 을 이루는 파티션들은 완벽히 같은 복제본이어야 하기 때문이다. 미러링이 작동되지 않을 경우, 피티션들중 하나를 RAID를 사용하지 않고 mount해서 사용하고, 아래와 같은 명령을 사용해 RAID 시스템을 복구한 후, 파티션을 unmount 하고, RAID 시스템을 다시 시작하여야 한다. 아래의 명령들은 RAID-1이 아닌 다른 레벨들에 사용하면 안된다는 것을 주의하라.

   Because each of the component partitions in a RAID-1 mirror is a perfectly valid copy of the file system. In a pinch, mirroring can be disabled, and one of the partitions can be mounted and safely run as an ordinary, non-RAID file system. When you are ready to restart using RAID-1, then unmount the partition, and follow the above instructions to restore the mirror. Note that the above works ONLY for RAID-1, and not for any of the other levels.

   위에서 처럼 망가지지 않은 파티션을 망가진 파티션으로 복사하는 것은 기분 좋은 일일 것이다. 이제 md 장치를 fsck 로 검사하기만 하면 된다.

   It may make you feel more comfortable to reverse the direction of the copy above: copy **from** the disk that was untouched **to** the one that was. Just be sure to fsck the final md.

7. **Q**: 나는 위의 질문들에 혼란스럽다. `fsck /dev/md0` 를 실행하는 것은 안전한가?

   A: 그렇다. `md장치들을 fsck 하는 것은 안전하다. 사실, 그게 안전하게 fsck를 실행시키는 유일한 방법이다. Yes, it is safe to run fsck on the md devices. In fact, this is the only safe place to run fsck.`

8. **Q**: 디스크가 천천히 오류나기 시작한다면, 어느 파티션의 오류인지 명백할것인가? 이런 혼란은 관리자로부터 위험한 결정을 내리게 할 수도 있지 않은가.

   A: 디스크에 문제가 생기기 새작하면, RAID의 저수준 드라이버가 error 코드를 반환할 것이다. RAID 드라이버는 좋은 쪽 disk의 superblock안에 "bad" 표시를 할것이고, 가능한 미러링을 유지하도록 명령할 것이다. (나중에 어떤 미러링이 좋은 쪽이고 나쁜쪽인지 배우게 될 것이다.) 물론, disk와 저수준 드라이버가 읽기/쓰기 에러를 감지할 것이고, 조용히 데이터가 망가지지는 않는다.

   Once a disk fails, an error code will be returned from the low level driver to the RAID driver. The RAID driver will mark it as "bad" in the RAID superblocks of the "good" disks (so we will later know which mirrors are good and which aren't), and continue RAID operation on the remaining operational mirrors.

   This, of course, assumes that the disk and the low level driver can detect a read/write error, and will not silently corrupt data, for example. This is true of current drives (error detection schemes are being used internally), and is the basis of RAID operation.

9. **Q**:

   hot-repair 는 무엇인가?

   A: RAID 시스템중 하나의 디스크가 망가졌을 때, RAID의 중단없이 실행중에 여분의 디스크의 추가를 통해 복구하는 "빠른 복구" 를 완성하려고 진행중이다. 그러나 이것을 사용하기 위해선, 여분의 디스크는 부팅시 선언되었거나. 몇몇 특별한 장비가 지원하는 전원이 들어온 상태에서 하드를 추가가 가능해야 한다.

   Work is underway to complete "hot reconstruction". With this feature, one can add several "spare" disks to the RAID set (be it level 1 or 4/5), and once a disk fails, it will be reconstructed on one of the spare disks in run time, without ever needing to shut down the array.

However, to use this feature, the spare disk must have been declared at boot time, or it must be hot-added, which requires the use of special cabinets and connectors that allow a disk to be added while the electrical power is on.

97년 10월 MD의 베타버젼이 할수 있는 것은 아래와 같다.

- 여분의 디스크를 통한 RAID 1 와 5 의 복구
- 잘못된 시스템종료시 RAID-5 parity 의 복구
- 작동하는 RAID 1, 4,5 시스템에 여분 드라이브 추가.

현재 기본적으로 자동복구는 설정되어 있지 않고, `include/linux/md.h` 안의 `SUPPORT_RECONSTRUCTION` 값을 바꾸어 설정할 수 있다.

As of October 97, there is a beta version of MD that allows:

- RAID 1 and 5 reconstruction on spare drives
- RAID-5 parity reconstruction after an unclean shutdown
- spare disk to be hot-added to an already running RAID 1 or 4/5 array

By default, automatic reconstruction is (Dec 97) currently disabled by default, due to the preliminary nature of this work. It can be enabled by changing the value of `SUPPORT_RECONSTRUCTION` in `include/linux/md.h`.

커널 기반의 복구가 설정되어 있고, RAID 시스템에 여분의 디스크(superblock은 이미 `mkraid`를 통해 만들어졌을 것이다.) 를 추가하려 한다면, 커널은 내용을 자동적으로 복구시켜 줄 것이다. (일반적인 `mdstop`, 디스크교체, `ckraid`, `mdrun` 의 절차를 밟지 않아도 된다.)

If spare drives were configured into the array when it was created and kernel-based reconstruction is enabled, the spare drive will already contain a RAID superblock (written by `mkraid`), and the kernel will reconstruct its contents automatically (without needing the usual `mdstop`, replace drive, `ckraid`, `mdrun` steps).

당신이 자동 복구를 실행하지 않았고, 교체할 디스크를 설정하지 않았다면, Gadi Oxman `<gadio@netvision.net.il>` 가 제안한 아래와 같은 단계를 따를 수 있다.

- 하나의 디스크가 제거되었다면, RAID는 degraged mode 로 설정되어 작동할 것이다. 이 것을 full operation mode로 장동시키기 위해서는 아래와 같은 절차가 필요하다.
  - RAID를 중단시켜라. (`mdstop /dev/md0`)
  - 고장난 디스크를 교체하라.
  - 내용복구를 위해 `ckraid raid.conf` 를 실행하라.
  - RAID를 다시실행시켜라. (`mdadd`, `mdrun`).

  중요한 점은 RAID는 다시 모든 드라이브에서 돌아갈 것이라는 것과. 하나의 디스크의 문제가 생겼을 때를 대비한 것이라는 것이다.

현재의 하나의 교체디스크를 여러개의 RAID에 배분하는 것은 불가능하다. 각각의 RAID는 각각의 disk를 필요로 한다.

If you are not running automatic reconstruction, and have not configured a hot-spare disk, the procedure described by Gadi Oxman `<gadio@netvision.net.il>` is recommended:

- Currently, once the first disk is removed, the RAID set will be running in degraded mode. To restore full operation mode, you need to:
  - stop the array (`mdstop /dev/md0`)
  - replace the failed drive
  - run `ckraid raid.conf` to reconstruct its contents
  - run the array again (`mdadd`, `mdrun`).

  At this point, the array will be running with all the drives, and again protects against a failure of a single drive.

Currently, it is not possible to assign single hot-spare disk to several arrays. Each array requires it's own hot-spare.

10. **Q**: 초보 관리자가 문제가 생겼다는 것을 알 수 있도록 "미러링되고 있는 디스크중 하나가 망가졌다.멍청아." 같은 경고를 소리로 들을 수 있기를 원한다.

    A: 커널은 "KERN_ALERT" 이벤트에 대해서 우선적으로 syslog에 로그를 남기고 있다. syslog를 모니터링할 몇몇 소프트웨어들이 있고, 그것들이 자동적으로 PC speaker로 beep를 울리거나, 삐삐를 호출하거나. e-mail등을 보낼것이다. The kernel is logging the event with a "KERN_ALERT" priority in syslog. There are several software packages that will monitor the syslog files, and beep the PC speaker, call a pager, send e-mail, etc. automatically.

11. **Q**: RAID-5를 어떻게 degraded mode로 사용할 수 있는가? (디스크 하나에 문제가 생겼고, 아직 교체하지 않았다.)

    A: Gadi Oxman <gadio@netvision.net.il> 이 적기를... 일반적으로, n 개의 드라이브로 raid-5 시스템을 돌리려면 아래와 같이 한다.:

    ```
    mdadd /dev/md0 /dev/disk1 ... /dev/disk(n)
    mdrun -p5 /dev/md0
    ```

    디스크중에 하나가 망가진 경우라도 여전히 mdadd를 사용해 설정해야 한다. (?? 망가진 디스크 대신 /dev/null을 사용해서 도전해라 ??? 조심해라..) (역자, 덧. 이 물음표는 뭘까... 이 문서의 저자는 이 문서의 윗부분에서 이 방법은 시도해 본적이 없다고 했다...-.-) RAID는 (n-1)개의 드라이브를 사용한 degraded mode로 동작할 것이다. "mdrun"가 실패했다면, kernel은 에러를 낼 것이다. (몇개의 문제가 있는 디스크라든지, shutdown을 제대로 안한 경우.) "dmesg" 명령어를 사용하여 kernel의 에러를 보아라. raid-5는 디스크가 하나 깨지는 것보다 전원이 나갔을 더 위험하며, 아래와 같이 새로운 RAID superblock를 만듦으로써 복구를 시도 할 수 있다.

    ```
    mkraid -f --only-superblock raid5.conf
    ```

    superblock의 복구는 모든 드라이브가 "OK" 로 표시되는 상태(아무일도 일어나지 않았다면.)에 영향을 받지 않기 때문에 간단할 것이다.

    Gadi Oxman <gadio@netvision.net.il> writes: Normally, to run a RAID-5 set of n drives you have to:

    ```
    mdadd /dev/md0 /dev/disk1 ... /dev/disk(n)
    mdrun -p5 /dev/md0
    ```

    Even if one of the disks has failed, you still have to mdadd it as you would in a normal setup. (?? try using /dev/null in place of the failed disk ??? watch out) Then,

    The array will be active in degraded mode with (n - 1) drives. If "mdrun" fails, the kernel has noticed an error (for example, several faulty drives, or an unclean shutdown). Use "dmesg" to display the kernel error messages from "mdrun". If the raid-5 set is corrupted due to a power loss, rather than a disk crash, one can try to recover by creating a new RAID superblock:

    ```
    mkraid -f --only-superblock raid5.conf
    ```

    A RAID array doesn't provide protection against a power failure or a kernel crash, and can't guarantee correct recovery. Rebuilding the superblock will simply cause the system to ignore the condition by marking all the drives as "OK", as if nothing happened.

12. **Q**: 디스크에 문제가 발생하면 RAID-5는 어떻게 동작하나요?

    A: 아래와 같은 전형적인 동작 단계가 있다.

    - RAID-5 가 작동한다.
    - RAID 작동중 하나의 디스크에 문제가 생겼다.

- 드라이브의 firmware와 저수준의 Linux 디스크 컨트롤러 드라이버는 오류를 감지하고 MD driver에 보고한다.
- MD driver는 나머지 사용가능한 드라이브들로, 커널의 상위레벨 부분에 에러와 관계없이 `/dev/md0` 를 제공할 것이다. (성능은 떨어진다.)
- 관리자는 일반적으로 `umount /dev/md0` 과 `mdstop /dev/md0`를 할 수 있다.
- 고장난 디스크가 교체되지 않아도, 관리자는 `mdadd` 와 `mdrun`를 실행시켜서. 여전히 degraded mode로 동작시킬 수 있을 것이다.

The typical operating scenario is as follows:

- A RAID-5 array is active.
- One drive fails while the array is active.
- The drive firmware and the low-level Linux disk/controller drivers detect the failure and report an error code to the MD driver.
- The MD driver continues to provide an error-free `/dev/md0` device to the higher levels of the kernel (with a performance degradation) by using the remaining operational drives.
- The sysadmin can `umount /dev/md0` and `mdstop /dev/md0` as usual.
- If the failed drive is not replaced, the sysadmin can still start the array in degraded mode as usual, by running `mdadd` and `mdrun`.

13. **Q**:

    A:

14. **Q**: 왜 13번째 질문은 없나요?

    A: 당신이, RAID와 높은 능력과 UPS와 관련이 있는 사람이라면, 그것들을 미신적으로 믿는 것 마저도 좋은 생각일 것이다. 그것은 절대 망가지지 않을 것이다. 그렇지 않은가? If you are concerned about RAID, High Availability, and UPS, then its probably a good idea to be superstitious as well. It can't hurt, can it?

15. **Q**: RAID-5 시스템에서 하나의 고장난 디스크를 교체했을 뿐인데.. RAID 를 복구한후 `fsck` 가 많은 error를 보여줍니다. 그게 정상인가요?

    A: 정상이 아니다. 그리고 fsck를 수정을 하지 않는 검사 전용모드에서 실행시키지 않았다면, 데이터에 문제가 생기는 것이 충분히 가능하다. 불행하게도, 디스크 교체후 RAID-5의 disk 순서를 우연히 바꾸어 버리는 흔한 실수 중의 하나이다. 비록 RAID superblock이 바람직한 방법으로 저장되긴 하지만, 모든 툴들이 이 정보를 따르는 것은 아니다. 특별히, ckraid 의 현재 버젼은 -f옵션을 사용해서 현재 superblock안의 데이터 대신, 정보를 읽어오도록 할 수 있다. (대체로 /etc/raid5.conf파일을 사용한다.) 지정한 정보가 부정확하면, 교체한 디스크가 부정확하게 복구될 것이고. 이런 종류의 실수들이 많은 fsck error들을 내는 증상을 보여준다. 그리고 당신이 신기한 경우에(이런 실수로 모든 데이터를 손실당하는..) 해당된다면, RAID의 재설정전에 모든 데이터를 백업하기를 강력히 추천한다.

    ```
    No.  And, unless you ran fsck in "verify only; do not update" mode,
    its quite possible that you have corrupted your data.  Unfortunately, a
    not-uncommon scenario is one of accidentally changing the disk order in a
    RAID-5 array, after replacing a hard drive.  Although the RAID superblock
    stores the proper order, not all tools use this information.  In particular,
    the current version of ckraid will use the information specified with the
    -f flag (typically, the file /etc/raid5.conf) instead of the data in the
    superblock.  If the specified order is incorrect, then the replaced disk
    will be reconstructed incorrectly.  The symptom of this kind of mistake
    ```

```
seems to be heavy & numerous fsck errors.  And, in case you are wondering,
yes, someone lost all of their data by making this mistake.  Making a
tape backup of all data before reconfiguring a RAID array is strongly
recommended.
```

16. **Q**: QuickStart 에서 `mdstop`는 단지 디스크들을 동기화(sync)시키는 것 뿐이라고 하는데, 그게 정말 필요한 가요? 파일시스템을 unmount하는 것으로 충분하지 않나요?

    A:
    `mdstop /dev/md0` 명령은.

    - shutdown이 잘 되었섰는지를 발견하기 위해, "clean"을 표시한다.
    - RAID를 동기화 시킨다. 후에 파일시스템의 unmount보다 중요하지 않지만, 파일시스템을 통하는 것이 아닌, `/dev/md0`을 직접 access 하기 때문에, 중요하다.

    The command `mdstop /dev/md0` will:

    - mark it "clean". This allows us to detect unclean shutdowns, for example due to a power failure or a kernel crash.
    - sync the array. This is less important after unmounting a filesystem, but is important if the `/dev/md0` is accessed directly rather than through a filesystem (for example, by `e2fsck`).

5 설치문제의 해결

1. **Q**:
   2.0.x 대의 커널을 위한 현재 알려진 가장 안정된 RAID는 패치는 무엇인가요?

   A:
   As of 18 Sept 1997, it is "2.0.30 + pre-9 2.0.31 + Werner Fink's swapping patch + the alpha RAID patch". As of November 1997, it is 2.0.31 + ... !?

2. **Q**: RAID 패치가 전 잘 인스톨되지 않네요. 무엇이 문제일까요?

   A: `/usr/include/linux` 를 `/usr/src/linux/include/linux`으로 심볼릭 링크를 걸어라. `raid5.c` 등의 새파일을 제대로 된 위치로 복사하라. 때때로 패치 명령어는 새로운 파일을 만들지 못한다. 패치시 `-f` 옵션을 사용해보라. Make sure that `/usr/include/linux` is a symbolic link to `/usr/src/linux/include/linux`.

   Make sure that the new files `raid5.c`, etc. have been copied to their correct locations. Sometimes the patch command will not create new files. Try the `-f` flag on `patch`.

3. **Q**: raidtools 0.42를 컴파일 중 include <pthread.h> 에서, 컴파일이 멎었다. 그게 내 시스템에는 없는데, 어떻게 이것을 고쳐야 하는가?

   A: raidtools-0.42는 `<ftp://ftp.inria.fr/INRIA/Projects/cristal/Xavier.Leroy>` 얻을 수 있는 linuxthreads-0.6을 필요로 한다. 또한, glibc v2.0도 사용한다. raidtools-0.42 requires linuxthreads-0.6 from: `<ftp://ftp.inria.fr/INRIA/Projects/cristal/Xavier.Leroy>` Alternately, use glibc v2.0.

4. **Q**: 이런 메시지가 나옵니다. `mdrun -a /dev/md0:  Invalid argument`

   A: 첫번째 사용하기 전에 `mkraid`를 사용하여 RAID를 초기화 하여야 한다. Use `mkraid` to initialize the RAID set prior to the first use. `mkraid` ensures that the RAID array is initially in a consistent state by erasing the RAID partitions. In addition, `mkraid` will create the RAID superblocks.

5. **Q**: 이런 메시지가 나옵니다. `mdrun -a /dev/md0: Invalid argument` 설정은.

   - RAID를 kernel 모듈로 설치하였고,
   - mdcreate, mdadd, 등의 순서대로 일반적인 인스톨을 했습니다.
   - `cat /proc/mdstat` 를 하면 아래와 같은 메시지가 나옵니다.

     ```
     Personalities :
     read_ahead not set
     md0 : inactive sda1 sdb1 6313482 blocks
     md1 : inactive
     md2 : inactive
     md3 : inactive
     ```

   - `mdrun -a` 로 오류를 출력해보면.. `/dev/md0: Invalid argument` 이런 메시지가 나옵니다.

   A: `lsmod` (또는 `cat /proc/modules`)로 raid 모듈이 로드되었는지 확인해라. 로드되어있지 않다면,`modprobe raid1` 또는 `modprobe raid5` 로 확실히 로드시켜라. 또는 autoloader를 사용한다면, `/etc/conf.modules` 에 아래와 같은 줄을 추가시켜라.

   ```
   alias md-personality-3 raid1
   alias md-personality-4 raid5
   ```

   Try `lsmod` (or, alternately, `cat /proc/modules`) to see if the raid modules are loaded. If they are not, you can load them explicitly with the `modprobe raid1` or `modprobe raid5` command. Alternately, if you are using the autoloader, and expected `kerneld` to load them and it didn't this is probably because your loader is missing the info to load the modules. Edit `/etc/conf.modules` and add the following lines:

   ```
   alias md-personality-3 raid1
   alias md-personality-4 raid5
   ```

6. **Q**: `mdadd -a` 의 실행중 `/dev/md0: No such file or directory` 와 같은 메시지가 나왔습니다. 정말 어디에도 `/dev/md0` 가 없습니다. 이것을 어떻게 해야 할까요?

   A: raid-tools는 루트가 `make install` 했을 때, 이 장치들을 만든다. 또한, 아래와 같이 할 수도 있다.

   ```
   cd /dev
   ./MAKEDEV md
   ```

   The raid-tools package will create these devices when you run `make install` as root. Alternately, you can do the following:

   ```
   cd /dev
   ./MAKEDEV md
   ```

7. **Q**: `/dev/md0`에 RAID를 만든 후 mount 시도시에 아래와 같은 에러가 납니다. 무엇이 문제입니까? `mount: wrong fs type, bad option, bad superblock on /dev/md0, or too many mounted file systems.`

   A: 마운트 하기전에 파일시스템을 만들어야 한다. `mke2fs`를 사용하라. You need to create a file system on `/dev/md0` before you can mount it. Use `mke2fs`.

8. **Q**: Truxton Fulton 작성: 제 2.0.30 시스템에서 `mkraid`로 RAID-1을 위해 각각의 파티션을 지우는 중. 콘솔상에 `"Cannot allocate free page"` 이런 에러가 나고, system log에 `"Unable to handle kernel paging request at virtual address ..."` 이런 에러가 납니다.

   Truxton Fulton wrote:

   > On my Linux 2.0.30 system, while doing a `mkraid` for a RAID-1 device, during the clearing of the two individual partitions, I got `"Cannot allocate free page"` errors appearing on the console, and `"Unable to handle kernel paging request at virtual address ..."` errors in the system log. At this time, the system became quite unusable, but it appears to recover after a while. The operation appears to have completed with no other errors, and I am successfully using my RAID-1 device. The errors are disconcerting though. Any ideas?

   > **A**: 그것은 2.0.30 kernel의 알려진 버그이다. 2.0.31로 수정하거나 2.0.29로 돌아가라. This was a well-known bug in the 2.0.30 kernels. It is fixed in the 2.0.31 kernel; alternately, fall back to 2.0.29.

9. **Q**: `mdadd` 된 장치를 `mdrun` 하려고 하면, 아래와 같은 메시지가 나옵니다. ”`invalid raid superblock magic`”.

   > **A**: `mkraid`를 실행하라. Make sure that you've run the `mkraid` part of the install procedure.

10. **Q**: 제가 `/dev/md0`을 사용하는 동안, 커널은 아래와 같은 에러들을 쏟아냅니다. `md0: device not running, giving up !` , `I/O error...` 전, 가상 device에 제 device를 성공적으로 추가시켰었습니다. When I access `/dev/md0`, the kernel spits out a lot of errors like `md0: device not running, giving up !` and `I/O error...`. I've successfully added my devices to the virtual device.

    > **A**: 사용하기 위해서 device는 실행되어야 한다. `mdrun -px /dev/md0`를 사용하라. x는 RAID level이다. To be usable, the device must be running. Use `mdrun -px /dev/md0` where x is l for linear, 0 for RAID-0 or 1 for RAID-1, etc.

11. **Q**: 두개의 device를 선형 연결해서 md device를 만들었습니다. `cat /proc/mdstat` 로는 총크기가 나오지만, `df` 명령으로는 첫번째 디스크의 크기밖에 안나옵니다.

    I've created a linear md-dev with 2 devices. `cat /proc/mdstat` shows the total size of the device, but `df` only shows the size of the first physical device.

    > **A**: 처음 사용하기 전에 반드시 `mkfs`를 실행야 한다. You must `mkfs` your new md-dev before using it the first time, so that the filesystem will cover the whole device.

12. **Q**: mdcreate로 `/etc/mdtab`를 설정하고 ,`mdadd`, `mdrun`,`fsck` 등을 사용해, 두개의 `/dev/mdX` 파티션을 만들었습니다. reboot 하기 전까지 모든 것이 괜찮아 보였지만, reboot 할때, 두 파티션에서 아래와 같은 `fsck` 가 났읍니다. `fsck.ext2: Attempt to read block from filesystem resulted in short read while trying too open /dev/md0` 왜 그렇고 어떻게 고쳐야 하나요? I've set up `/etc/mdtab` using mdcreate, I've `mdadd`'ed, `mdrun` and `fsck`'ed my two `/dev/mdX` partitions. Everything looks okay before a reboot. As soon as I reboot, I get an `fsck` error on both partitions: `fsck.ext2: Attempt to read block from filesystem resulted in short read while trying too open /dev/md0`. Why?! How do I fix it?!

    > **A**: 부팅 작업시 RAID 파티션들은 `fsck`이전에 반드시 시작되어야 한다. `fsck` 는 `/etc/rc.d/rc.S` 또는 `/etc/rc.d/rc.sysinit` 에서 불려질 것이다. 이 파일들의 `fsck -A` 전에 `mdadd -ar`를 추가해라.

During the boot process, the RAID partitions must be started before they can be `fsck`'ed. This must be done in one of the boot scripts. For some distributions, `fsck` is called from `/etc/rc.d/rc.S`, for others, it is called from `/etc/rc.d/rc.sysinit`. Change this file to `mdadd -ar` *before* `fsck -A` is executed. Better yet, it is suggested that `ckraid` be run if `mdadd` returns with an error. How do do this is discussed in greater detail in question 14 of the section "Error Recovery".

13. **Q**: 4GB 보다 큰 파티션들을 포함하는 RAID를 실행시키려 도전할 때 아래와 같은 메시지가 나왔습니다. `invalid raid superblock magic`

    A: 이 버그는 현재 고쳐졌다.(97년 9월) 최근의 raid버져을 사용해라. This bug is now fixed. (September 97) Make sure you have the latest raid code.

14. **Q**: 2GB보다 큰 파티션으로 mke2fs를 시도하는데 아래와 같은 오류가 납니다. `Warning: could not write 8 blocks in inode table starting at 2097175`

    A: 이것은 `mke2fs`의 문제 같다. 임시로 소스에서 `e2fsprogs-1.10/lib/ext2fs/llseek.c` 파일의 첫 `#ifdef HAVE_LLSEEK` 전에 `#undef HAVE_LLSEEK` 를 추가한후 mke2fs를 재컴파일해서 사용하라. This seems to be a problem with `mke2fs` (November 97). A temporary work-around is to get the mke2fs code, and add `#undef HAVE_LLSEEK` to `e2fsprogs-1.10/lib/ext2fs/llseek.c` just before the first `#ifdef HAVE_LLSEEK` and recompile mke2fs.

15. **Q**: `ckraid` 가 `/etc/mdtab` 를 읽지 못합니다.

    A: `/etc/mdtab` 안의 RAID-0 / linear RAID 설정은 쓰이지 않고 있고, 좀더 후에야 지원될것이다. `/etc/raid1.conf` 등의 설정파일들을 사용하라. The RAID0/linear configuration file format used in `/etc/mdtab` is obsolete, although it will be supported for a while more. The current, up-to-date config files are currently named `/etc/raid1.conf`, etc.

16. **Q**: (`raid1.o`) 같은 모듈들이 자동으로 로드되지 않습니다. mdrun 전에 수동으로 modprobe를 실행시켜야 하는데, 어떻게 이것을 고쳐야 할까요?

    A: `/etc/conf.modules`에 아래와 같은 줄을 추가해라.

    ```
    alias md-personality-3 raid1
    alias md-personality-4 raid5
    ```

    To autoload the modules, we can add the following to `/etc/conf.modules`:

    ```
    alias md-personality-3 raid1
    alias md-personality-4 raid5
    ```

17. **Q**: 드라이브 13개를 `mdadd` 한 후, `mdrun -p5 /dev/md0` 시도 했지만, 아래와 같은 오류가 나왔습니다. `/dev/md0: Invalid argument`

    A: software RAID의 기본설정 8개이다. `linux/md.h` 의 `#define MAX_REAL=8` 를 크게 바꾼후, 커널을 재컴파일 하라. The default configuration for software RAID is 8 real devices. Edit `linux/md.h`, change `#define MAX_REAL=8` to a larger number, and rebuild the kernel.

18. **Q**: 최신식 SPARCstation 5 에서 md 작업을 할수 없습니다. 디스크 라벨에 무슨 문제가 있는 것이라고 생각됩니다만.. I can't make `md` work with partitions on our latest SPARCstation 5. I suspect that this has something to do with disk-labels.

A: Sun 의 디스크 라벨은 파티션의 첫 1K 에 있다. RAID-1에서는 ext2fs 파티션이 모든 라벨을 무시 할 것이기 때문에 문제가 없지만, 다른 레벨들에 관해서는 아직 해결되지 않았다. Sun disk-labels sit in the first 1K of a partition. For RAID-1, the Sun disk-label is not an issue since `ext2fs` will skip the label on every mirror. For other raid levels (0, linear and 4/5), this appears to be a problem; it has not yet (Dec 97) been addressed.

6 Hardware & Software 의 지원

1. **Q**: 각각 다른 회사의 디스크와 SCSI adapter 로 RAID를 구성할 수 있는가? I have SCSI adapter brand XYZ (with or without several channels), and disk brand(s) PQR and LMN, will these work with md to create a linear/stripped/mirrored personality?

   A: 물론이다. 소프트웨어 RAID는 어떤 디스크와 디스크콘트롤러에서도 작동한다, 예를 들면, SCSI와 IDE를 섞어서 RAID 를 만들수도 있다. 디스크 크기가 같을 필요도 없으며, 그 외 어떤 제약도 없다. 이것은 디스크 그대로가 아닌, 파티션을 사용하기 때문에 가능하다. 단지, RAID 레벨 1과 5의 경우 같은 크기의 파티션을 사용할 것을 권장할 뿐이다. 같은 크기의 파티션을 사용하지 않으면 남는 파티션은 버려지게 된다.

   Yes! Software RAID will work with any disk controller (IDE or SCSI) and any disks. The disks do not have to be identical, nor do the controllers. For example, a RAID mirror can be created with one half the mirror being a SCSI disk, and the other an IDE disk. The disks do not even have to be the same size. There are no restrictions on the mixing & matching of disks and controllers.

   This is because Software RAID works with disk partitions, not with the raw disks themselves. The only recommendation is that for RAID levels 1 and 5, the disk partitions that are used as part of the same set be the same size. If the partitions used to make up the RAID 1 or 5 array are not the same size, then the excess space in the larger partitions is wasted (not used).

2. **Q**: 나는 하드웨어 RAID 0,1,0+1을 지원하는 두 채널짜리 BT-952를 가지고 있다.... I have a twin channel BT-952, and the box states that it supports hardware RAID 0, 1 and 0+1. I have made a RAID set with two drives, the card apparently recognizes them when it's doing it's BIOS startup routine. I've been reading in the driver source code, but found no reference to the hardware RAID support. Anybody out there working on that?

   A: RAIDPlus 를 지원하는 Mylex/BusLogic FlashPoint 보드들은, 전부 하드웨어를 사용하는 것이 아니라, 실제로 소프트웨어 RAID도 사용하고, 단지 Windows 95 and Windows NT에서만 지원된다. The Mylex/BusLogic FlashPoint boards with RAIDPlus are actually software RAID, not hardware RAID at all. RAIDPlus is only supported on Windows 95 and Windows NT, not on Netware or any of the Unix platforms. Aside from booting and configuration, the RAID support is actually in the OS drivers.

   While in theory Linux support for RAIDPlus is possible, the implementation of RAID-0/1/4/5 in the Linux kernel is much more flexible and should have superior performance, so there's little reason to support RAIDPlus directly.

3. **Q**: SMP box에서 RAID를 사용하는 것이 안전한가? I want to run RAID with an SMP box. Is RAID SMP-safe?

   A: "나는 그렇게 생각한다." 가 내가 지금 할수 있는 가장 좋은 답이다. 최근에 많은 유저들이 SMP에서의 RAID에 대해보고하고 있다. 하지만, 아래와 같은 문제가 메일링 리스트를 통해 넘어오기도 했다. "I think so" is the best answer available at the time I write this (April 98). A number of users report that they have been using RAID with SMP for nearly a year, without problems. However, as of April 98 (circa kernel 2.1.9x), the following problems have been noted on the mailing list:

- Adaptec AIC7xxx SCSI drivers are not SMP safe (General note: Adaptec adapters have a long & lengthly history of problems & flakiness in general. Although they seem to be the most easily available, widespread and cheapest SCSI adapters, they should be avoided. After factoring for time lost, frustration, and corrupted data, Adaptec's will prove to be the costliest mistake you'll ever make. That said, if you have SMP problems with 2.1.88, try the patch ftp://ftp.bero-online.ml.org/pub/linux/aic7xxx-5.0.7-linux21.tar.gz I am not sure if this patch has been pulled into later 2.1.x kernels. For further info, take a look at the mail archives for March 98 at http://www.linuxhq.com/lnxlists/linux-raid/lr_9803_01/ As usual, due to the rapidly changing nature of the latest experimental 2.1.x kernels, the problems described in these mailing lists may or may not have been fixed by the time your read this. Caveat Emptor.)
- IO-APIC with RAID-0 on SMP has been reported to crash in 2.1.90

7 이미 설치된 시스템 수정하기.

1. **Q**: 선형 RAID가 확장가능한가? 디스크를 추가해서 존재하는 파일 시스템의 크기를 늘릴 수 있는가? Are linear MD's expandable? Can a new hard-drive/partition be added, and the size of the existing file system expanded?

   A: Miguel de Icaza <`miguel@luthien.nuclecu.unam.mx`> 는 아래와 같이 말했다.

   나는 파일 시스템 당 하나의 디스크대신 여러개의 디스크를 사용할 경우를 대비해서 ext2fs의 소스코드를 고쳤다. 그래서, 파일 시스템을 늘리고자 한다면, 유틸리티로 새디스크를 적당히 설정후 다만, 시스템에 알리기만 하면된다. 시스템이 작동중에 곧바로 추가가 가능하고, 재부팅할 필요는 없다. 아래의 홈페이지에서, 2.1.x버젼대의 커널패치를 받아라.

   `<http://www.nuclecu.unam.mx/miguel/ext2-volume>`

   Miguel de Icaza <`miguel@luthien.nuclecu.unam.mx`> writes:

   I changed the ext2fs code to be aware of multiple-devices instead of the regular one device per file system assumption.

   So, when you want to extend a file system, you run a utility program that makes the appropriate changes on the new device (your extra partition) and then you just tell the system to extend the fs using the specified device.

   You can extend a file system with new devices at system operation time, no need to bring the system down (and whenever I get some extra time, you will be able to remove devices from the ext2 volume set, again without even having to go to single-user mode or any hack like that).

   You can get the patch for 2.1.x kernel from my web page:

   `<http://www.nuclecu.unam.mx/miguel/ext2-volume>`

2. **Q**: RAID-5 시스템에 디스크를 추가할 수 있는가?

   A: 현재, 모든 데이터를 지우지 않고는 불가능하다. 컨버젼하는 도구는 아직 없으며, RAID-5의 실제적 구조는, disk의 수에 의존하기 때문이다. 물론, 모든 데이터를 백업한후, 시스템을 다시 구축하면 가능하다. Currently, (September 1997) no, not without erasing all data. A conversion utility to allow this does not yet exist. The problem is that the actual structure and layout of a RAID-5 array depends on the number of disks in the array.

Of course, one can add drives by backing up the array to tape, deleting all data, creating a new array, and restoring from tape.

3. **Q**: RAID1/RAID0 으로 사용하고 있는 `/dev/hdb` 를 `/dev/hdc`로 이동시키려 합니다. 만약, `/etc/mdtab` 과 `/etc/raid1.conf`의 설정만 바꾼다고 한다면, 어떤 일이 일어날까요? What would happen to my RAID1/RAID0 sets if I shift one of the drives from being `/dev/hdb` to `/dev/hdc`?

   Because of cabling/case size/stupidity issues, I had to make my RAID sets on the same IDE controller (`/dev/hda` and `/dev/hdb`). Now that I've fixed some stuff, I want to move `/dev/hdb` to `/dev/hdc`.

   What would happen if I just change the `/etc/mdtab` and `/etc/raid1.conf` files to reflect the new location?

   A: linear 와 RAID-0에서는 정확히 같은 명령으로 드라이브를 지정해야 한다. 예를 들면 원래 설정이, 아래와 같다면,

   ```
   mdadd /dev/md0 /dev/hda /dev/hdb
   ```

   새로운 설정은 반드시 아래와 같아야 할 것이다.

   ```
   mdadd /dev/md0 /dev/hda /dev/hdc
   ```

   RAID-1/4/5에서는 superblock에 "RAID number"가 저장되어 있기 때문에, 어떤 디스크를 지정하는가는 별로 중요하지 않다.

   For RAID-0/linear, one must be careful to specify the drives in exactly the same order. Thus, in the above example, if the original config is

   ```
   mdadd /dev/md0 /dev/hda /dev/hdb
   ```

   Then the new config *must* be

   ```
   mdadd /dev/md0 /dev/hda /dev/hdc
   ```

   For RAID-1/4/5, the drive's "RAID number" is stored in its RAID superblock, and therefore the order in which the disks are specified is not important.

   RAID-0/linear does not have a superblock due to it's older design, and the desire to maintain backwards compatibility with this older design.

4. **Q**: 디스크 두개를 사용하는 RAID-1 시스템을 세개의 디스크를 사용하는 RAID-5 시스템으로 바꿀수 있는가?

   A: 할수 있다. BizSystems 의 Michael은 이걸 쉽게 하는 방법을 만들어냈다. 그러나, 실수로 인해 데이터를 날릴 수 있으므로, 필히 백업해놓기를 바란다.

   Yes. Michael at BizSystems has come up with a clever, sneaky way of doing this. However, like virtually all manipulations of RAID arrays once they have data on them, it is dangerous and prone to human error. **Make a backup before you start**.

   ```
   I will make the following assumptions:
   ---------------------------------------------
   disks
   original: hda - hdc
   raid1 partitions hda3 - hdc3
   array name /dev/md0
   ```

```
new hda - hdc - hdd
raid5 partitions hda3 - hdc3 - hdd3
array name: /dev/md1

You must substitute the appropriate disk and partition numbers for
you system configuration. This will hold true for all config file
examples.
--------------------------------------------
DO A BACKUP BEFORE YOU DO ANYTHING
1) recompile kernel to include both raid1 and raid5
2) install new kernel and verify that raid personalities are present
3) disable the redundant partition on the raid 1 array. If this is a
 root mounted partition (mine was) you must be more careful.

 Reboot the kernel without starting raid devices or boot from rescue
 system ( raid tools must be available )

 start non-redundant raid1
mdadd -r -p1 /dev/md0 /dev/hda3

4) configure raid5 but with 'funny' config file, note that there is
  no hda3 entry and hdc3 is repeated. This is needed since the
  raid tools don't want you to do this.
-------------------------------
# raid-5 configuration
raiddev                 /dev/md1
raid-level              5
nr-raid-disks           3
chunk-size              32

# Parity placement algorithm
parity-algorithm        left-symmetric

# Spare disks for hot reconstruction
nr-spare-disks          0

device                  /dev/hdc3
raid-disk               0

device                  /dev/hdc3
raid-disk               1

device                  /dev/hdd3
raid-disk               2
---------------------------------------
 mkraid /etc/raid5.conf
5) activate the raid5 array in non-redundant mode

mdadd -r -p5 -c32k /dev/md1 /dev/hdc3 /dev/hdd3
```

```
6) make a file system on the array

mke2fs -b {blocksize} /dev/md1

recommended blocksize by some is 4096 rather than the default 1024.
this improves the memory utilization for the kernel raid routines and
matches the blocksize to the page size. I compromised and used 2048
since I have a relatively high number of small files on my system.

7) mount the two raid devices somewhere

mount -t ext2 /dev/md0 mnt0
mount -t ext2 /dev/md1 mnt1

8) move the data

cp -a mnt0 mnt1

9) verify that the data sets are identical
10) stop both arrays
11) correct the information for the raid5.conf file
  change /dev/md1 to /dev/md0
  change the first disk to read /dev/hda3

12) upgrade the new array to full redundant status
 (THIS DESTROYS REMAINING raid1 INFORMATION)

ckraid --fix /etc/raid5.conf
```

8 성능과 도구, 일반적인 것들에 관한 멍청한 질문들.

1. **Q**:

   I've created a RAID-0 device on `/dev/sda2` and `/dev/sda3`. The device is a lot slower than a single partition. Isn't md a pile of junk?

   > **A**:
   > To have a RAID-0 device running a full speed, you must have partitions from different disks. Besides, putting the two halves of the mirror on the same disk fails to give you any protection whatsoever against disk failure.

2. **Q**:

   What's the use of having RAID-linear when RAID-0 will do the same thing, but provide higher performance?

   > **A**:
   > It's not obvious that RAID-0 will always provide better performance; in fact, in some cases, it could make things worse. The ext2fs file system scatters files all over a partition, and it attempts to keep all of the blocks of a file contiguous, basically in an attempt to

prevent fragmentation. Thus, ext2fs behaves "as if" there were a (variable-sized) stripe per file. If there are several disks concatenated into a single RAID-linear, this will result files being statistically distributed on each of the disks. Thus, at least for ext2fs, RAID-linear will behave a lot like RAID-0 with large stripe sizes. Conversely, RAID-0 with small stripe sizes can cause excessive disk activity leading to severely degraded performance if several large files are accessed simultaneously. In many cases, RAID-0 can be an obvious win. For example, imagine a large database file. Since ext2fs attempts to cluster together all of the blocks of a file, chances are good that it will end up on only one drive if RAID-linear is used, but will get chopped into lots of stripes if RAID-0 is used. Now imagine a number of (kernel) threads all trying to random access to this database. Under RAID-linear, all accesses would go to one disk, which would not be as efficient as the parallel accesses that RAID-0 entails.

3. **Q**:

How does RAID-0 handle a situation where the different stripe partitions are different sizes? Are the stripes uniformly distributed?

A: To understand this, lets look at an example with three partitions; one that is 50MB, one 90MB and one 125MB.

Lets call D0 the 50MB disk, D1 the 90MB disk and D2 the 125MB disk. When you start the device, the driver calculates 'strip zones'. In this case, it finds 3 zones, defined like this:

```
Z0 : (D0/D1/D2) 3 x 50 = 150MB  total in this zone
Z1 : (D1/D2)  2 x 40 = 80MB total in this zone
Z2 : (D2) 125-50-40 = 35MB total in this zone.
```

You can see that the total size of the zones is the size of the virtual device, but, depending on the zone, the striping is different. Z2 is rather inefficient, since there's only one disk.

Since `ext2fs` and most other Unix file systems distribute files all over the disk, you have a $35/265 = 13\%$ chance that a fill will end up on Z2, and not get any of the benefits of striping.

(DOS tries to fill a disk from beginning to end, and thus, the oldest files would end up on Z0. However, this strategy leads to severe filesystem fragmentation, which is why no one besides DOS does it this way.)

4. **Q**: I have some Brand X hard disks and a Brand Y controller. and am considering using `md`. Does it significantly increase the throughput? Is the performance really noticeable?

A: The answer depends on the configuration that you use.

Linux MD RAID-0 and RAID-linear performance:
If the system is heavily loaded with lots of I/O, statistically, some of it will go to one disk, and some to the others. Thus, performance will improve over a single large disk. The actual improvement depends a lot on the actual data, stripe sizes, and other factors. In a system with low I/O usage, the performance is equal to that of a single disk.

Linux MD RAID-1 (mirroring) read performance:
MD implements read balancing. That is, the RAID-1 code will alternate between each of the (two or more) disks in the mirror, making alternate reads to each. In a low-I/O situation, this won't change performance at all: you will have to wait for one disk to complete the read. But, with two disks in a high-I/O environment, this could as much as double the read performance, since reads can be issued to each of the disks in parallel. For N disks in the mirror, this could improve performance N-fold.

Linux MD RAID-1 (mirroring) write performance:
Must wait for the write to occur to all of the disks in the mirror. This is because a copy of the data must be written to each of the disks in the mirror. Thus, performance will be roughly equal to the write performance to a single disk.

Linux MD RAID-4/5 read performance:
Statistically, a given block can be on any one of a number of disk drives, and thus RAID-4/5 read performance is a lot like that for RAID-0. It will depend on the data, the stripe size, and the application. It will not be as good as the read performance of a mirrored array.

Linux MD RAID-4/5 write performance:
This will in general be considerably slower than that for a single disk. This is because the parity must be written out to one drive as well as the data to another. However, in order to compute the new parity, the old parity and the old data must be read first. The old data, new data and old parity must all be XOR'ed together to determine the new parity: this requires considerable CPU cycles in addition to the numerous disk accesses.

5. **Q**: What RAID configuration should I use for optimal performance?

   A: Is the goal to maximize throughput, or to minimize latency? There is no easy answer, as there are many factors that affect performance:

   - operating system - will one process/thread, or many be performing disk access?
   - application - is it accessing data in a sequential fashion, or random access?
   - file system - clusters files or spreads them out (the ext2fs clusters together the blocks of a file, and spreads out files)
   - disk driver - number of blocks to read ahead (this is a tunable parameter)
   - CEC hardware - one drive controller, or many?
   - hd controller - able to queue multiple requests or not? Does it provide a cache?
   - hard drive - buffer cache memory size – is it big enough to handle the write sizes and rate you want?
   - physical platters - blocks per cylinder – accessing blocks on different cylinders will lead to seeks.

6. **Q**: What is the optimal RAID-5 configuration for performance?

   A: Since RAID-5 experiences an I/O load that is equally distributed across several drives, the best performance will be obtained when the RAID set is balanced by using identical drives, identical controllers, and the same (low) number of drives on each controller.

   Note, however, that using identical components will raise the probability of multiple simultaneous failures, for example due to a sudden jolt or drop, overheating, or a power surge during an electrical storm. Mixing brands and models helps reduce this risk.

7. **Q**: What is the optimal block size for a RAID-4/5 array?

   A: When using the current (November 1997) RAID-4/5 implementation, it is strongly recommended that the file system be created with `mke2fs -b 4096` instead of the default 1024 byte filesystem block size.

   This is because the current RAID-5 implementation allocates one 4K memory page per disk block; if a disk block were just 1K in size, then 75% of the memory which RAID-5 is allocating for pending I/O would not be used. If the disk block size matches the memory page size, then the driver can (potentially) use all of the page. Thus, for a filesystem with a 4096 block size as opposed to a 1024 byte block size, the RAID driver will potentially queue 4 times as much pending I/O to the low level drivers without allocating additional memory.

Note: the above remarks do NOT apply to Software RAID-0/1/linear driver.

Note: the statements about 4K memory page size apply to the Intel x86 architecture. The page size on Alpha, Sparc, and other CPUS are different; I believe they're 8K on Alpha/Sparc (????). Adjust the above figures accordingly.

Note: if your file system has a lot of small files (files less than 10KBytes in size), a considerable fraction of the disk space might be wasted. This is because the file system allocates disk space in multiples of the block size. Allocating large blocks for small files clearly results in a waste of disk space: thus, you may want to stick to small block sizes, get a larger effective storage capacity, and not worry about the "wasted" memory due to the block-size/page-size mismatch.

Note: most "typical" systems do not have that many small files. That is, although there might be thousands of small files, this would lead to only some 10 to 100MB wasted space, which is probably an acceptable tradeoff for performance on a multi-gigabyte disk.

However, for news servers, there might be tens or hundreds of thousands of small files. In such cases, the smaller block size, and thus the improved storage capacity, may be more important than the more efficient I/O scheduling.

Note: there exists an experimental file system for Linux which packs small files and file chunks onto a single block. It apparently has some very positive performance implications when the average file size is much smaller than the block size.

Note: Future versions may implement schemes that obsolete the above discussion. However, this is difficult to implement, since dynamic run-time allocation can lead to dead-locks; the current implementation performs a static pre-allocation.

8. **Q**: How does the chunk size (stripe size) influence the speed of my RAID-0, RAID-4 or RAID-5 device?

A: The chunk size is the amount of data contiguous on the virtual device that is also contiguous on the physical device. In this HOWTO, "chunk" and "stripe" refer to the same thing: what is commonly called the "stripe" in other RAID documentation is called the "chunk" in the MD man pages. Stripes or chunks apply only to RAID 0, 4 and 5, since stripes are not used in mirroring (RAID-1) and simple concatenation (RAID-linear). The stripe size affects both read and write latency (delay), throughput (bandwidth), and contention between independent operations (ability to simultaneously service overlapping I/O requests). Assuming the use of the ext2fs file system, and the current kernel policies about read-ahead, large stripe sizes are almost always better than small stripe sizes, and stripe sizes from about a fourth to a full disk cylinder in size may be best. To understand this claim, let us consider the effects of large stripes on small files, and small stripes on large files. The stripe size does not affect the read performance of small files: For an array of N drives, the file has a 1/N probability of being entirely within one stripe on any one of the drives. Thus, both the read latency and bandwidth will be comparable to that of a single drive. Assuming that the small files are statistically well distributed around the filesystem, (and, with the ext2fs file system, they should be), roughly N times more overlapping, concurrent reads should be possible without significant collision between them. Conversely, if very small stripes are used, and a large file is read sequentially, then a read will issued to all of the disks in the array. For a the read of a single large file, the latency will almost double, as the probability of a block being 3/4'ths of a revolution or farther away will increase. Note, however, the trade-off: the bandwidth could improve almost N-fold for reading a single, large file, as N drives can be reading simultaneously (that is, if read-ahead is used so that all of the disks are kept active). But there is another, counter-acting trade-off: if all of the drives are already busy reading one file, then attempting to read a second or third file at the same time will cause significant contention, ruining performance as the disk ladder algorithms lead to seeks all over the platter. Thus, large stripes will almost always lead to the best performance.

The sole exception is the case where one is streaming a single, large file at a time, and one requires the top possible bandwidth, and one is also using a good read-ahead algorithm, in which case small stripes are desired.

Note that this HOWTO previously recommended small stripe sizes for news spools or other systems with lots of small files. This was bad advice, and here's why: news spools contain not only many small files, but also large summary files, as well as large directories. If the summary file is larger than the stripe size, reading it will cause many disks to be accessed, slowing things down as each disk performs a seek. Similarly, the current ext2fs file system searches directories in a linear, sequential fashion. Thus, to find a given file or inode, on average half of the directory will be read. If this directory is spread across several stripes (several disks), the directory read (e.g. due to the ls command) could get very slow. Thanks to Steven A. Reisman <`sar@pressenter.com`> for this correction. Steve also adds:

> I found that using a 256k stripe gives much better performance. I suspect that the optimum size would be the size of a disk cylinder (or maybe the size of the disk drive's sector cache). However, disks nowadays have recording zones with different sector counts (and sector caches vary among different disk models). There's no way to guarantee stripes won't cross a cylinder boundary.

The tools accept the stripe size specified in KBytes. You'll want to specify a multiple of if the page size for your CPU (4KB on the x86).

9. **Q**: What is the correct stride factor to use when creating the ext2fs file system on the RAID partition? By stride, I mean the -R flag on the `mke2fs` command:

```
mke2fs -b 4096 -R stride=nnn  ...
```

What should the value of nnn be?

A: The `-R stride` flag is used to tell the file system about the size of the RAID stripes. Since only RAID-0,4 and 5 use stripes, and RAID-1 (mirroring) and RAID-linear do not, this flag is applicable only for RAID-0,4,5.

Knowledge of the size of a stripe allows `mke2fs` to allocate the block and inode bitmaps so that they don't all end up on the same physical drive. An unknown contributor wrote:

> I noticed last spring that one drive in a pair always had a larger I/O count, and tracked it down to the these meta-data blocks. Ted added the `-R stride=` option in response to my explanation and request for a workaround.

For a 4KB block file system, with stripe size 256KB, one would use `-R stride=64`. If you don't trust the `-R` flag, you can get a similar effect in a different way. Steven A. Reisman <`sar@pressenter.com`> writes:

> Another consideration is the filesystem used on the RAID-0 device. The ext2 filesystem allocates 8192 blocks per group. Each group has its own set of inodes. If there are 2, 4 or 8 drives, these inodes cluster on the first disk. I've distributed the inodes across all drives by telling mke2fs to allocate only 7932 blocks per group.

Some mke2fs pages do not describe the `[-g blocks-per-group]` flag used in this operation.

10. **Q**: Where can I put the `md` commands in the startup scripts, so that everything will start automatically at boot time?

A: Rod Wilkens <`rwilkens@border.net`> writes:

What I did is put "`mdadd -ar`" in the "`/etc/rc.d/rc.sysinit`" right after the kernel loads the modules, and before the "`fsck`" disk check. This way, you can put the "`/dev/md?`" device in the "`/etc/fstab`". Then I put the "`mdstop -a`" right after the "`umount -a`" unmounting the disks, in the "`/etc/rc.d/init.d/halt`" file. For raid-5, you will want to look at the return code for `mdadd`, and if it failed, do a

```
ckraid --fix /etc/raid5.conf
```

to repair any damage.

11. **Q**: I was wondering if it's possible to setup striping with more than 2 devices in `md0`? This is for a news server, and I have 9 drives... Needless to say I need much more than two. Is this possible?

    A: Yes. (describe how to do this)

12. **Q**: When is Software RAID superior to Hardware RAID?

    A: Normally, Hardware RAID is considered superior to Software RAID, because hardware controllers often have a large cache, and can do a better job of scheduling operations in parallel. However, integrated Software RAID can (and does) gain certain advantages from being close to the operating system.

    For example, ... ummm. Opaque description of caching of reconstructed blocks in buffer cache elided ...

    On a dual PPro SMP system, it has been reported that Software-RAID performance exceeds the performance of a well-known hardware-RAID board vendor by a factor of 2 to 5.

    Software RAID is also a very interesting option for high-availability redundant server systems. In such a configuration, two CPU's are attached to one set or SCSI disks. If one server crashes or fails to respond, then the other server can `mdadd`, `mdrun` and `mount` the software RAID array, and take over operations. This sort of dual-ended operation is not always possible with many hardware RAID controllers, because of the state configuration that the hardware controllers maintain.

13. **Q**: If I upgrade my version of raidtools, will it have trouble manipulating older raid arrays? In short, should I recreate my RAID arrays when upgrading the raid utilities?

    A: No, not unless the major version number changes. An MD version x.y.z consists of three sub-versions:

    ```
    x:      Major version.
    y:      Minor version.
    z:      Patchlevel version.
    ```

    Version x1.y1.z1 of the RAID driver supports a RAID array with version x2.y2.z2 in case (x1 == x2) and (y1 >= y2).

    Different patchlevel (z) versions for the same (x.y) version are designed to be mostly compatible.

    The minor version number is increased whenever the RAID array layout is changed in a way which is incompatible with older versions of the driver. New versions of the driver will maintain compatibility with older RAID arrays.

    The major version number will be increased if it will no longer make sense to support old RAID arrays in the new kernel code.

    For RAID-1, it's not likely that the disk layout nor the superblock structure will change anytime soon. Most all Any optimization and new features (reconstruction, multithreaded tools, hot-plug, etc.) doesn't affect the physical layout.

14. **Q**: The command `mdstop /dev/md0` says that the device is busy.

    A: There's a process that has a file open on `/dev/md0`, or `/dev/md0` is still mounted. Terminate the process or `umount /dev/md0`.

15. **Q**: Are there performance tools?

    A: There is also a new utility called `iotrace` in the `linux/iotrace` directory. It reads `/proc/io-trace` and analyses/plots it's output. If you feel your system's block IO performance is too low, just look at the iotrace output.

16. **Q**: I was reading the RAID source, and saw the value `SPEED_LIMIT` defined as 1024K/sec. What does this mean? Does this limit performance?

    A: `SPEED_LIMIT` is used to limit RAID reconstruction speed during automatic reconstruction. Basically, automatic reconstruction allows you to `e2fsck` and `mount` immediately after an unclean shutdown, without first running `ckraid`. Automatic reconstruction is also used after a failed hard drive has been replaced.

    In order to avoid overwhelming the system while reconstruction is occurring, the reconstruction thread monitors the reconstruction speed and slows it down if its too fast. The 1M/sec limit was arbitrarily chosen as a reasonable rate which allows the reconstruction to finish reasonably rapidly, while creating only a light load on the system so that other processes are not interfered with.

17. **Q**: What about "spindle synchronization" or "disk synchronization"?

    A: Spindle synchronization is used to keep multiple hard drives spinning at exactly the same speed, so that their disk platters are always perfectly aligned. This is used by some hardware controllers to better organize disk writes. However, for software RAID, this information is not used, and spindle synchronization might even hurt performance.

18. **Q**: How can I set up swap spaces using raid 0? Wouldn't striped swap ares over 4+ drives be really fast?

    A: Leonard N. Zubkoff replies: It is really fast, but you don't need to use MD to get striped swap. The kernel automatically stripes across equal priority swap spaces. For example, the following entries from `/etc/fstab` stripe swap space across five drives in three groups:

    ```
    /dev/sdg1       swap    swap    pri=3
    /dev/sdk1       swap    swap    pri=3
    /dev/sdd1       swap    swap    pri=3
    /dev/sdh1       swap    swap    pri=3
    /dev/sdl1       swap    swap    pri=3
    /dev/sdg2       swap    swap    pri=2
    /dev/sdk2       swap    swap    pri=2
    /dev/sdd2       swap    swap    pri=2
    /dev/sdh2       swap    swap    pri=2
    /dev/sdl2       swap    swap    pri=2
    /dev/sdg3       swap    swap    pri=1
    /dev/sdk3       swap    swap    pri=1
    /dev/sdd3       swap    swap    pri=1
    /dev/sdh3       swap    swap    pri=1
    /dev/sdl3       swap    swap    pri=1
    ```

19. **Q**: I want to maximize performance. Should I use multiple controllers?

A: In many cases, the answer is yes. Using several controllers to perform disk access in parallel will improve performance. However, the actual improvement depends on your actual configuration. For example, it has been reported (Vaughan Pratt, January 98) that a single 4.3GB Cheetah attached to an Adaptec 2940UW can achieve a rate of 14MB/sec (without using RAID). Installing two disks on one controller, and using a RAID-0 configuration results in a measured performance of 27 MB/sec.

Note that the 2940UW controller is an "Ultra-Wide" SCSI controller, capable of a theoretical burst rate of 40MB/sec, and so the above measurements are not surprising. However, a slower controller attached to two fast disks would be the bottleneck. Note also, that most out-board SCSI enclosures (e.g. the kind with hot-pluggable trays) cannot be run at the 40MB/sec rate, due to cabling and electrical noise problems.

If you are designing a multiple controller system, remember that most disks and controllers typically run at 70-85% of their rated max speeds.

Note also that using one controller per disk can reduce the likelihood of system outage due to a controller or cable failure (In theory – only if the device driver for the controller can gracefully handle a broken controller. Not all SCSI device drivers seem to be able to handle such a situation without panicking or otherwise locking up).

9 고 가용성(High-Availability) RAID

1. **Q**: RAID는 데이터 손실을 막아줄 수 있다, 하지만, 시스템을 손상없이 가능한 오래 켜놓을 수 있는가?

   RAID can help protect me against data loss. But how can I also ensure that the system is up as long as possible, and not prone to breakdown? Ideally, I want a system that is up 24 hours a day, 7 days a week, 365 days a year.

   A: 고 가용성(High-Availability)은 좀 더 어렵고 비싼 것이다. 아래의 hint와 tip, 생각, 소문이 그 문제에 대해서 도와줄 것이다.

   - 같은 IDE 리본 케이블에 연결된 디스크중 하나에 오류가 나면, 디스크 두개가 모두 망가진 것으로 인식될 것이다. 하나의 IDE 케이블에는 하나의 디스크만을 사용해라.
   - SCSI chain 역시 하나의 오류디스크가 모든 디스크에 접근 못하게 할것이다. 같은 SCSI chain에 같은 RAID 시스템의 디스크들을 두지 말아라.
   - 디스크 콘트롤러도 역시 여러개를 사용하라.
   - 모든 디스크를 같은 회사, 같은 모델로 쓰지 말아라. 디스크들이 물리적인 충격을 받을 경우 좀 더 안전할 것이다.
   - CPU나 콘트롤러의 실패할 경우에 대비해서 SCSI를 두개의 컴퓨터와 연결되는 "twin-tailed" 상태로 설정할 수 있을 것이다. (아래 원문을 참고 하시길.. -.-; 쩝.)
   - 항상 UPS를 사용하고 shutdown을 하라.
   - SCSI 케이블은 매우 까다롭고, 문제가 되기 쉬운 것으로 알려져 있다. 살 수 있는 가장 좋은 질의 케이블을 사용해라.
   - SSI (Serial Storage Architecture) 를 보고 다소 비싸더라도 안전하다고 알려진 제품을 사용해라.
   - 즐겨라, 망가지는 것은 당신이 생각하는 것보다 나중 일일 것이다.

   High-Availability is difficult and expensive. The harder you try to make a system be fault tolerant, the harder and more expensive it gets. The following hints, tips, ideas and unsubstantiated rumors may help you with this quest.

   - IDE disks can fail in such a way that the failed disk on an IDE ribbon can also prevent the good disk on the same ribbon from responding, thus making it look as if two disks have failed. Since

RAID does not protect against two-disk failures, one should either put only one disk on an IDE cable, or if there are two disks, they should belong to different RAID sets.

- SCSI disks can fail in such a way that the failed disk on a SCSI chain can prevent any device on the chain from being accessed. The failure mode involves a short of the common (shared) device ready pin; since this pin is shared, no arbitration can occur until the short is removed. Thus, no two disks on the same SCSI chain should belong to the same RAID array.
- Similar remarks apply to the disk controllers. Don't load up the channels on one controller; use multiple controllers.
- Don't use the same brand or model number for all of the disks. It is not uncommon for severe electrical storms to take out two or more disks. (Yes, we all use surge suppressors, but these are not perfect either). Heat & poor ventilation of the disk enclosure are other disk killers. Cheap disks often run hot. Using different brands of disk & controller decreases the likelihood that whatever took out one disk (heat, physical shock, vibration, electrical surge) will also damage the others on the same date.
- To guard against controller or CPU failure, it should be possible to build a SCSI disk enclosure that is "twin-tailed": i.e. is connected to two computers. One computer will mount the file-systems read-write, while the second computer will mount them read-only, and act as a hot spare. When the hot-spare is able to determine that the master has failed (e.g. through a watchdog), it will cut the power to the master (to make sure that it's really off), and then fsck & remount read-write. If anyone gets this working, let me know.
- Always use an UPS, and perform clean shutdowns. Although an unclean shutdown may not damage the disks, running ckraid on even small-ish arrays is painfully slow. You want to avoid running ckraid as much as possible. Or you can hack on the kernel and get the hot-reconstruction code debugged ...
- SCSI cables are well-known to be very temperamental creatures, and prone to cause all sorts of problems. Use the highest quality cabling that you can find for sale. Use e.g. bubble-wrap to make sure that ribbon cables to not get too close to one another and cross-talk. Rigorously observe cable-length restrictions.
- Take a look at SSI (Serial Storage Architecture). Although it is rather expensive, it is rumored to be less prone to the failure modes that SCSI exhibits.
- Enjoy yourself, its later than you think.

10 답변을 기다리는 질문들.

1. **Q**: If, for cost reasons, I try to mirror a slow disk with a fast disk, is the S/W smart enough to balance the reads accordingly or will it all slow down to the speed of the slowest?
2. **Q**: For testing the raw disk thru put... is there a character device for raw read/raw writes instead of `/dev/sdaxx` that we can use to measure performance on the raid drives?? is there a GUI based tool to use to watch the disk thru-put??

11 추가되어지기 바라는 기능들..

Bradley Ward Allen <`ulmo@Q.Net`> wrote:

Ideas include:

- Boot-up parameters to tell the kernel which devices are to be MD devices (no more "`mdadd`")

- Making MD transparent to "`mount`"/"`umount`" such that there is no "`mdrun`" and "`mdstop`"
- Integrating "`ckraid`" entirely into the kernel, and letting it run as needed

(So far, all I've done is suggest getting rid of the tools and putting them into the kernel; that's how I feel about it, this is a filesystem, not a toy.)

- Deal with arrays that can easily survive N disks going out simultaneously or at separate moments, where N is a whole number > 0 settable by the administrator
- Handle kernel freezes, power outages, and other abrupt shutdowns better
- Don't disable a whole disk if only parts of it have failed, e.g., if the sector errors are confined to less than 50% of access over the attempts of 20 dissimilar requests, then it continues just ignoring those sectors of that particular disk.
- Bad sectors:
  - A mechanism for saving which sectors are bad, someplace onto the disk.
  - If there is a generalized mechanism for marking degraded bad blocks that upper filesystem levels can recognize, use that. Program it if not.
  - Perhaps alternatively a mechanism for telling the upper layer that the size of the disk got smaller, even arranging for the upper layer to move out stuff from the areas being eliminated. This would help with a degraded blocks as well.
  - Failing the above ideas, keeping a small (admin settable) amount of space aside for bad blocks (distributed evenly across disk?), and using them (nearby if possible) instead of the bad blocks when it does happen. Of course, this is inefficient. Furthermore, the kernel ought to log every time the RAID array starts each bad sector and what is being done about it with a "`crit`" level warning, just to get the administrator to realize that his disk has a piece of dust burrowing into it (or a head with platter sickness).
- Software-switchable disks:

  "disable this disk"
  would block until kernel has completed making sure there is no data on the disk being shut down that is needed (e.g., to complete an XOR/ECC/other error correction), then release the disk from use (so it could be removed, etc.);

  "enable this disk"
  would `mkraid` a new disk if appropriate and then start using it for ECC/whatever operations, enlarging the RAID5 array as it goes;

  "resize array"
  would respecify the total number of disks and the number of redundant disks, and the result would often be to resize the size of the array; where no data loss would result, doing this as needed would be nice, but I have a hard time figuring out how it would do that; in any case, a mode where it would block (for possibly hours (kernel ought to log something every ten seconds if so)) would be necessary;

  "enable this disk while saving data"
  which would save the data on a disk as-is and move it to the RAID5 system as needed, so that a horrific save and restore would not have to happen every time someone brings up a RAID5 system (instead, it may be simpler to only save one partition instead of two, it might fit onto the first as a gzip'd file even); finally,

  "re-enable disk"
  would be an operator's hint to the OS to try out a previously failed disk (it would simply call disable then enable, I suppose).

Other ideas off the net:

  - finalrd analog to initrd, to simplify root raid.
- a read-only raid mode, to simplify the above
- Mark the RAID set as clean whenever there are no "half writes" done. – That is, whenever there are no write transactions that were committed on one disk but still unfinished on another disk.

  Add a "write inactivity" timeout (to avoid frequent seeks to the RAID superblock when the RAID set is relatively busy).