Chapter 3
Data Protection: RAID

KEY CONCEPTS

Hardware and Software RAID

Striping, Mirroring, and Parity

RAID Levels

RAID Write Penalty

Hot Spares

In the late 1980s, rapid adoption of computers for business processes stimulated the growth of new applications and databases, significantly increasing the demand for storage capacity and performance. At that time, data was stored on a single large, expensive disk drive called *Single Large Expensive Drive* (SLED). Use of single disks could not meet the required performance levels because they were capable of serving only a limited number of I/Os.

Today's data centers house hundreds of disk drives in their storage infrastructure. Disk drives are inherently susceptible to failures due to mechanical wear and tear and other environmental factors, which could result in data loss. The greater the number of disk drives in a storage array, the greater the probability of a disk failure in the array. For example, consider a storage array of 100 disk drives, each with an average life expectancy of 750,000 hours. The average life expectancy of this collection in the array, therefore, is 750,000/100 or 7,500 hours. This means that a disk drive in this array is likely to fail at least once in 7,500 hours.

RAID is an enabling technology that leverages multiple drives as part of a set that provides data protection against drive failures. In general, RAID implementations also improve the storage system performance by serving I/Os from multiple disks simultaneously. Modern arrays with flash drives also benefit in terms of protection and performance by using RAID.

In 1987, Patterson, Gibson, and Katz at the University of California, Berkeley, published a paper titled "A Case for Redundant Arrays of Inexpensive Disks

(RAID)." This paper described the use of small-capacity, inexpensive disk drives as an alternative to large-capacity drives common on mainframe computers. The term *RAID* has been redefined to refer to *independent* disks to reflect advances in the storage technology. RAID technology has now grown from an academic concept to an industry standard and is common implementation in today's storage arrays.

This chapter details RAID technology, RAID levels, and different types of RAID implementations and their benefits.

3.1 RAID Implementation Methods

The two methods of RAID implementation are hardware and software. Both have their advantages and disadvantages, and are discussed in this section.

3.1.1 Software RAID

Software RAID uses host-based software to provide RAID functions. It is implemented at the operating-system level and does not use a dedicated hardware controller to manage the RAID array.

Software RAID implementations offer cost and simplicity benefits when compared with hardware RAID. However, they have the following limitations:

- **Performance:** Software RAID affects overall system performance. This is due to additional CPU cycles required to perform RAID calculations.
- **Supported features:** Software RAID does not support all RAID levels.
- **Operating system compatibility:** Software RAID is tied to the host operating system; hence, upgrades to software RAID or to the operating system should be validated for compatibility. This leads to inflexibility in the data-processing environment.

3.1.2 Hardware RAID

In *hardware RAID* implementations, a specialized hardware controller is implemented either on the host or on the array.

Controller card RAID is a host-based hardware RAID implementation in which a specialized RAID controller is installed in the host, and disk drives are connected to it. Manufacturers also integrate RAID controllers on motherboards. A host-based RAID controller is not an efficient solution in a data center environment with a large number of hosts.

The external RAID controller is an array-based hardware RAID. It acts as an interface between the host and disks. It presents storage volumes to the host, and the host manages these volumes as physical drives. The key functions of the RAID controllers are as follows:

- Management and control of disk aggregations
- Translation of I/O requests between logical disks and physical disks
- Data regeneration in the event of disk failures

3.2 RAID Array Components

A *RAID array* is an enclosure that contains a number of disk drives and supporting hardware to implement RAID. A subset of disks within a RAID array can be grouped to form logical associations called logical arrays, also known as a *RAID set* or a *RAID group* (see Figure 3-1).

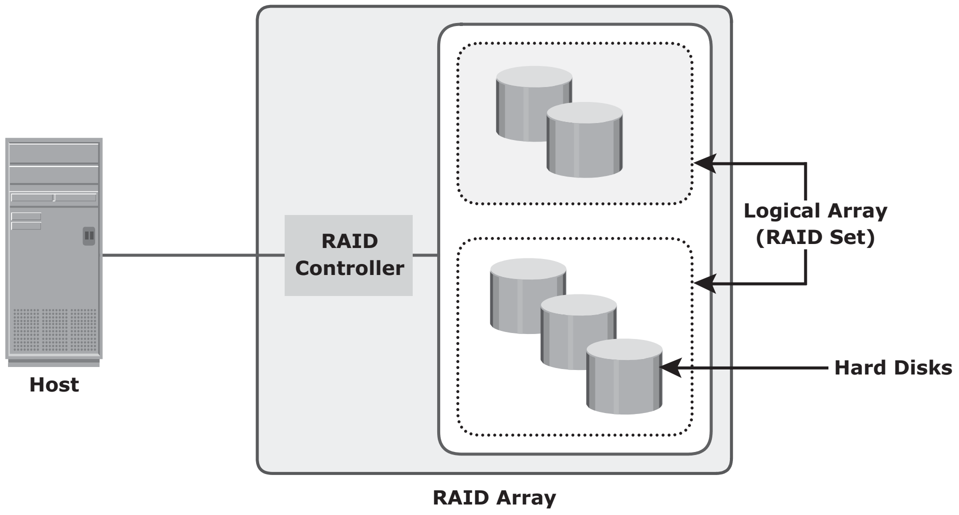

Figure 3-1: Components of a RAID array

3.3 RAID Techniques

RAID techniques — striping, mirroring, and parity — form the basis for defining various RAID levels. These techniques determine the data availability and performance characteristics of a RAID set.

3.3.1 Striping

Striping is a technique to spread data across multiple drives (more than one) to use the drives in parallel. All the read-write heads work simultaneously, allowing more data to be processed in a shorter time and increasing performance, compared to reading and writing from a single disk.

Within each disk in a RAID set, a predefined number of contiguously addressable disk blocks are defined as a *strip*. The set of aligned strips that spans across all the disks within the RAID set is called a *stripe*. Figure 3-2 shows physical and logical representations of a striped RAID set.

Strip size (also called *stripe depth*) describes the number of blocks in a strip and is the maximum amount of data that can be written to or read from a single disk in the set, assuming that the accessed data starts at the beginning of the strip. All strips in a stripe have the same number of blocks. Having a smaller strip size means that data is broken into smaller pieces while spread across the disks.

Stripe size is a multiple of strip size by the number of *data* disks in the RAID set. For example, in a five disk striped RAID set with a strip size of 64 KB, the stripe size is 320 KB(64KB × 5). *Stripe width* refers to the number of data strips in a stripe. Striped RAID does not provide any data protection unless parity or mirroring is used, as discussed in the following sections.

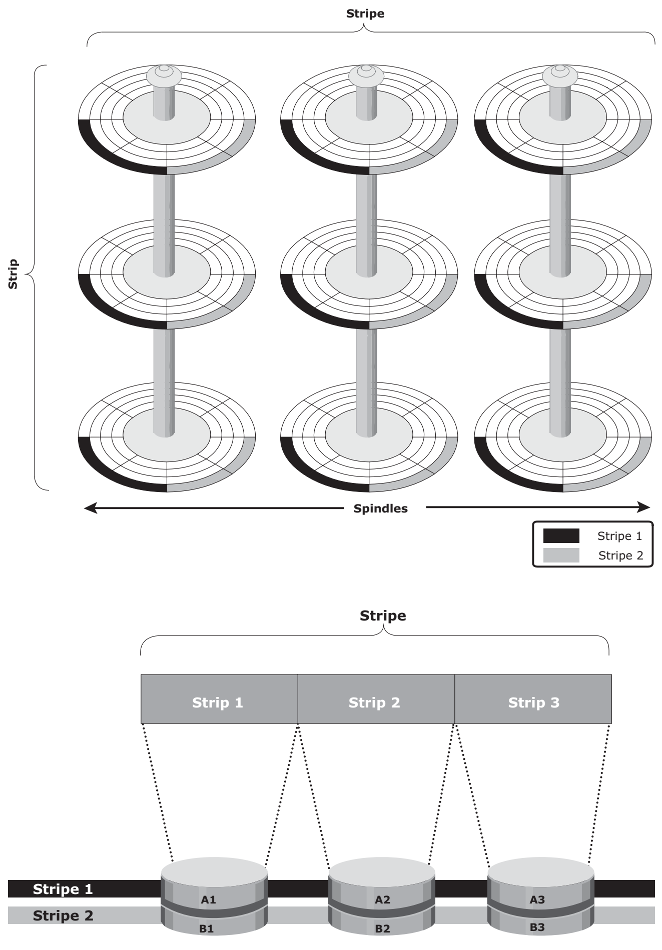

Figure 3-2: Striped RAID set

3.3.2 Mirroring

Mirroring is a technique whereby the same data is stored on two different disk drives, yielding two copies of the data. If one disk drive failure occurs, the data is intact on the surviving disk drive (see Figure 3-3) and the controller continues to service the host's data requests from the surviving disk of a mirrored pair.

When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair. This activity is transparent to the host.

In addition to providing complete data redundancy, mirroring enables fast recovery from disk failure. However, disk mirroring provides only data protection and is not a substitute for data backup. Mirroring constantly captures changes in the data, whereas a backup captures point-in-time images of the data.

Mirroring involves duplication of data — the amount of storage capacity needed is twice the amount of data being stored. Therefore, mirroring is considered expensive and is preferred for mission-critical applications that cannot afford the risk of any data loss. Mirroring improves read performance because read requests can be serviced by both disks. However, write performance is slightly lower than that in a single disk because each write request manifests as two writes on the disk drives. Mirroring does not deliver the same levels of write performance as a striped RAID.

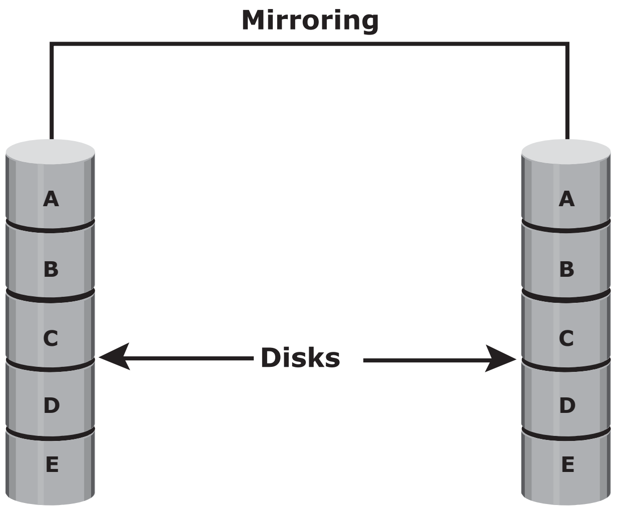

Figure 3-3: Mirrored disks in an array

3.3.3 Parity

Parity is a method to protect striped data from disk drive failure without the cost of mirroring. An additional disk drive is added to hold parity, a mathematical construct that allows re-creation of the missing data. Parity is a redundancy technique that ensures protection of data without maintaining a full set of duplicate data. Calculation of parity is a function of the RAID controller.

Parity information can be stored on separate, dedicated disk drives or distributed across all the drives in a RAID set. Figure 3-4 shows a parity RAID set. The first four disks, labeled "Data Disks," contain the data. The fifth disk, labeled "Parity Disk," stores the parity information, which, in this case, is the sum of the elements in each row. Now, if one of the data disks fails, the missing value can be calculated by subtracting the sum of the rest of the elements from the parity value. Here, for simplicity, the computation of parity is represented as an arithmetic sum of the data. However, parity calculation is a *bitwise XOR* operation.

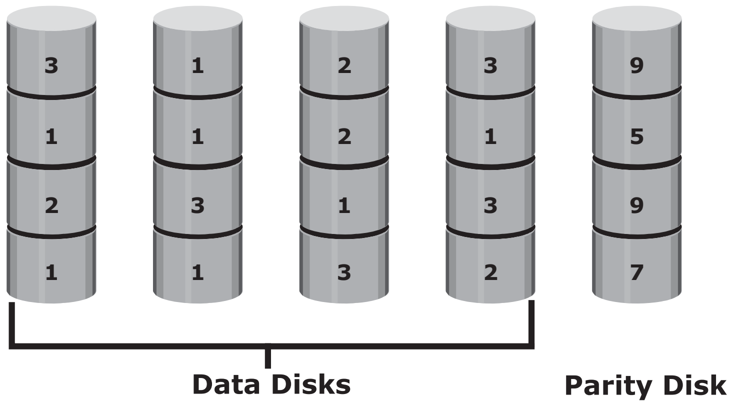

Figure 3-4: Parity RAID

XOR OPERATION

A bit-by-bit Exclusive -OR (XOR) operation takes two bit patterns of equal length and performs the logical XOR operation on each pair of corresponding bits. The result in each position is 1 if the two bits are different, and 0 if they are the same. The truth table of the XOR operation is shown next. (A and B denote the inputs and C, the output after performing the XOR operation.) If any of the data from A, B, or C is lost, it can be reproduced by performing an XOR operation on the remaining available data. For example, if a disk containing all the data from A fails, the data can be regenerated by performing an XOR between B and C.

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

Compared to mirroring, parity implementation considerably reduces the cost associated with data protection. Consider an example of a parity RAID configuration with five disks where four disks hold data, and the fifth holds the parity information. In this example, parity requires only 25 percent extra disk space compared to mirroring, which requires 100 percent extra disk space. However, there are some disadvantages of using parity. Parity information is generated from data on the data disk. Therefore, parity is recalculated every time there is a change in data. This recalculation is time-consuming and affects the performance of the RAID array.

For parity RAID, the stripe size calculation does not include the parity strip. For example in a five (4 + 1) disk parity RAID set with a strip size of 64 KB, the stripe size will be 256 KB (64 KB × 4).

3.4 RAID Levels

Application performance, data availability requirements, and cost determine the RAID level selection. These RAID levels are defined on the basis of striping, mirroring, and parity techniques. Some RAID levels use a single technique, whereas others use a combination of techniques. Table 3-1 shows the commonly used RAID levels.

Table 3-1: Raid Levels

LEVELS	BRIEF DESCRIPTION
RAID 0	Striped set with no fault tolerance
RAID 1	Disk mirroring
Nested	Combinations of RAID levels. Example: RAID 1 + RAID 0
RAID 3	Striped set with parallel access and a dedicated parity disk
RAID 4	Striped set with independent disk access and a dedicated parity disk
RAID 5	Striped set with independent disk access and distributed parity
RAID 6	Striped set with independent disk access and dual distributed parity

3.4.1 RAID 0

RAID 0 configuration uses data striping techniques, where data is striped across all the disks within a RAID set. Therefore it utilizes the full storage capacity of a RAID set. To read data, all the strips are put back together by the controller. Figure 3-5 shows RAID 0 in an array in which data is striped across five disks. When the number of drives in the RAID set increases, performance improves because more data can be read or written simultaneously. RAID 0 is a good option for applications that need high I/O throughput. However, if these applications require high availability during drive failures, RAID 0 does not provide data protection and availability.

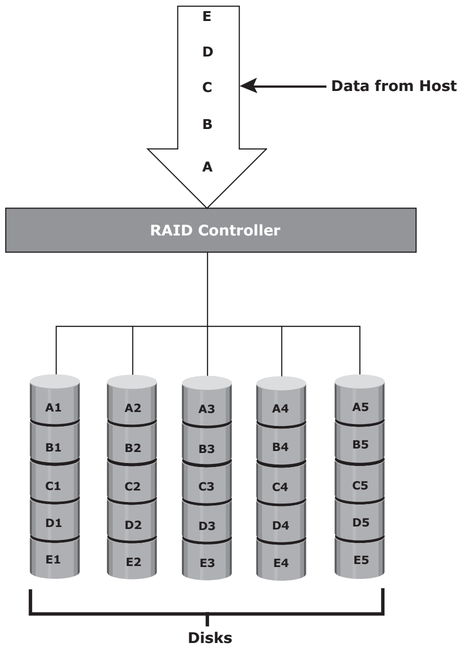

Figure 3-5: RAID 0

3.4.2 RAID 1

RAID 1 is based on the mirroring technique. In this RAID configuration, data is mirrored to provide fault tolerance (see Figure 3-6). A RAID 1 set consists of two disk drives and every write is written to both disks. The mirroring is transparent to the host. During disk failure, the impact on data recovery in RAID 1 is the least among all RAID implementations. This is because the RAID controller uses the mirror drive for data recovery. RAID 1 is suitable for applications that require high availability and cost is no constraint.

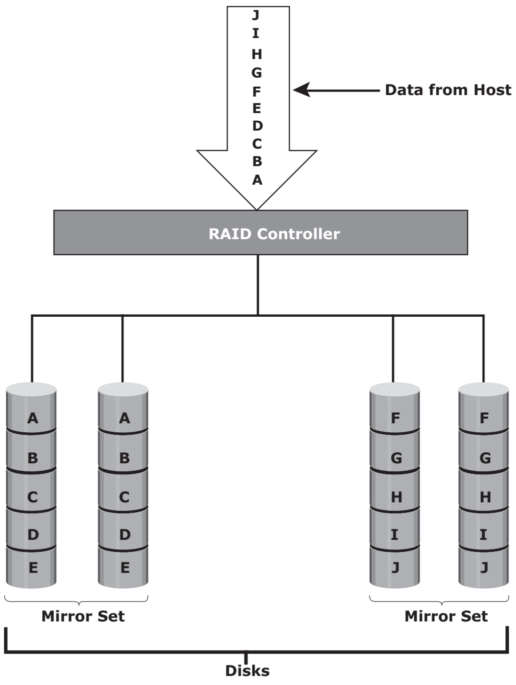

Figure 3-6: RAID 1

3.4.3 Nested RAID

Most data centers require data redundancy and performance from their RAID arrays. RAID 1+0 and RAID 0+1combine the performance benefits of RAID 0 with the redundancy benefits of RAID 1. They use striping and mirroring techniques and combine their benefits. These types of RAID require an even number of disks, the minimum being four (see Figure 3-7).

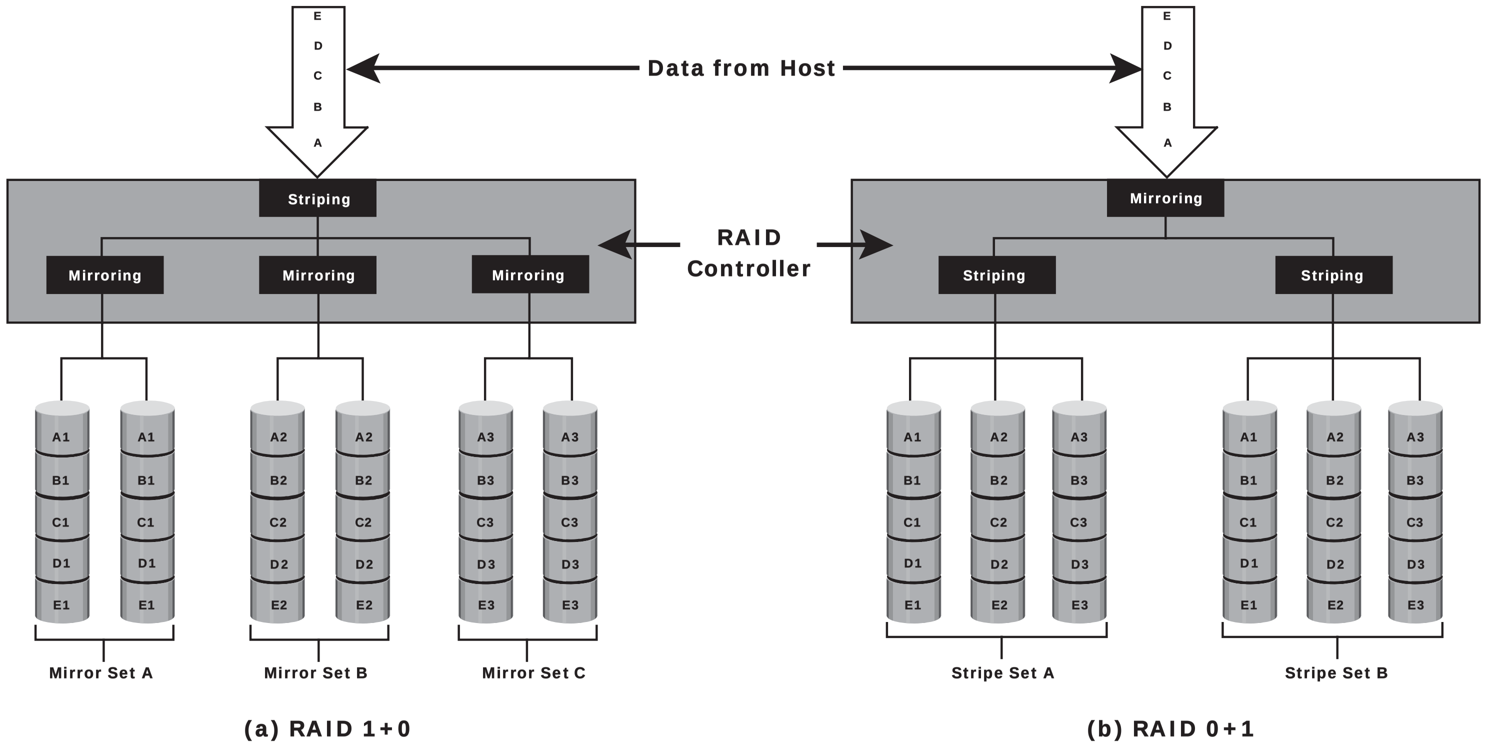

Figure 3-7: Nested RAID

RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0. Similarly, RAID 0+1 is also known as RAID 01 or RAID 0/1. RAID 1+0 performs well for workloads with small, random, write-intensive I/Os. Some applications that benefit from RAID 1+0 include the following:

- High transaction rate Online Transaction Processing (OLTP)
- Large messaging installations
- Database applications with write intensive random access workloads

A common misconception is that RAID 1+0 and RAID 0+1 are the same. Under normal conditions, RAID levels 1+0 and 0+1 offer identical benefits. However, rebuild operations in the case of disk failure differ between the two.

RAID 1+0 is also called striped mirror. The basic element of RAID 1+0 is a mirrored pair, which means that data is first mirrored and then both copies of the data are striped across multiple disk drive pairs in a RAID set. When replacing a failed drive, only the mirror is rebuilt. In other words, the disk array controller uses the surviving drive in the mirrored pair for data recovery and continuous operation. Data from the surviving disk is copied to the replacement disk.

To understand the working of RAID 1+0, consider an example of six disks forming a RAID 1+0 (RAID 1 first and then RAID 0) set. These six disks are paired into three sets of two disks, where each set acts as a RAID 1 set (mirrored pair of disks). Data is then striped across all the three mirrored sets to form RAID 0. Following are the steps performed in RAID 1+0 (see Figure 3-7 [a]):

Drives 1+2 = RAID 1 (Mirror Set A)
Drives 3+4 = RAID 1 (Mirror Set B)
Drives 5+6 = RAID 1 (Mirror Set C)

Now, RAID 0 striping is performed across sets A through C. In this configuration, if drive 5 fails, then the mirror set C alone is affected. It still has drive 6 and continues to function and the entire RAID 1+0 array also keeps functioning. Now, suppose drive 3 fails while drive 5 was being replaced. In this case the array still continues to function because drive 3 is in a different mirror set. So, in this configuration, up to three drives can fail without affecting the array, as long as they are all in different mirror sets.

RAID 0+1 is also called a mirrored stripe. The basic element of RAID 0+1 is a stripe. This means that the process of striping data across disk drives is performed initially, and then the entire stripe is mirrored. In this configuration if one drive fails, then the entire stripe is faulted. Consider the same example of six disks to understand the working of RAID 0+1 (that is, RAID 0 first and then RAID 1). Here, six disks are paired into two sets of three disks each. Each of these sets, in turn, act as a RAID 0 set that contains three disks and then these two sets are mirrored to form RAID 1. Following are the steps performed in RAID 0+1 (see Figure 3-7 [b]):

Drives 1 + 2 + 3 = RAID 0 (Stripe Set A)
Drives 4 + 5 + 6 = RAID 0 (Stripe Set B)

Now, these two stripe sets are mirrored. If one of the drives, say drive 3, fails, the entire stripe set A fails. A rebuild operation copies the entire stripe, copying the data from each disk in the healthy stripe to an equivalent disk in the failed stripe. This causes increased and unnecessary I/O load on the surviving disks and makes the RAID set more vulnerable to a second disk failure.

3.4.4 RAID 3

RAID 3 stripes data for performance and uses parity for fault tolerance. Parity information is stored on a dedicated drive so that the data can be reconstructed if a drive fails in a RAID set. For example, in a set of five disks, four are used for data and one for parity. Therefore, the total disk space required is 1.25 times the size of the data disks. RAID 3 always reads and writes complete stripes of data across all disks because the drives operate in parallel. There are no partial writes that update one out of many strips in a stripe. Figure 3-8 illustrates the RAID 3 implementation.

RAID 3 provides good performance for applications that involve large sequential data access, such as data backup or video streaming.

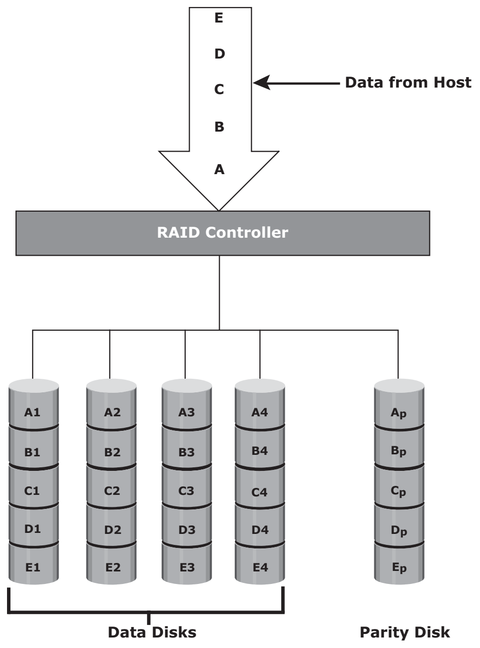

Figure 3-8: RAID 3

3.4.5 RAID 4

Similar to RAID 3, RAID 4 stripes data for high performance and uses parity for improved fault tolerance. Data is striped across all disks except the parity disk in the array. Parity information is stored on a dedicated disk so that the data can be rebuilt if a drive fails.

Unlike RAID 3, data disks in RAID 4 can be accessed independently so that specific data elements can be read or written on a single disk without reading or writing an entire stripe. RAID 4 provides good read throughput and reasonable write throughput.

3.4.6 RAID 5

RAID 5 is a versatile RAID implementation. It is similar to RAID 4 because it uses striping. The drives (strips) are also independently accessible. The difference between RAID 4 and RAID 5 is the parity location. In RAID 4, parity is written to a dedicated drive, creating a write bottleneck for the parity disk. In RAID 5, parity is distributed across all disks to overcome the write bottleneck of a dedicated parity disk. Figure 3-9 illustrates the RAID 5 implementation.

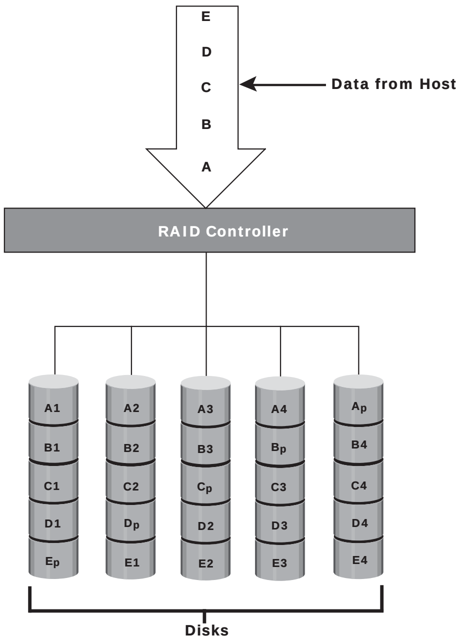

Figure 3-9: RAID 5

RAID 5 is good for random, read-intensive I/O applications and preferred for messaging, data mining, medium-performance media serving, and relational database management system (RDBMS) implementations, in which database administrators (DBAs) optimize data access.

3.4.7 RAID 6

RAID 6 works the same way as RAID 5, except that RAID 6 includes a second parity element to enable survival if two disk failures occur in a RAID set (see Figure 3-10). Therefore, a RAID 6 implementation requires at least four disks. RAID 6 distributes the parity across all the disks. The write penalty (explained later in this chapter) in RAID 6 is more than that in RAID 5; therefore, RAID 5 writes perform better than RAID 6. The rebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.

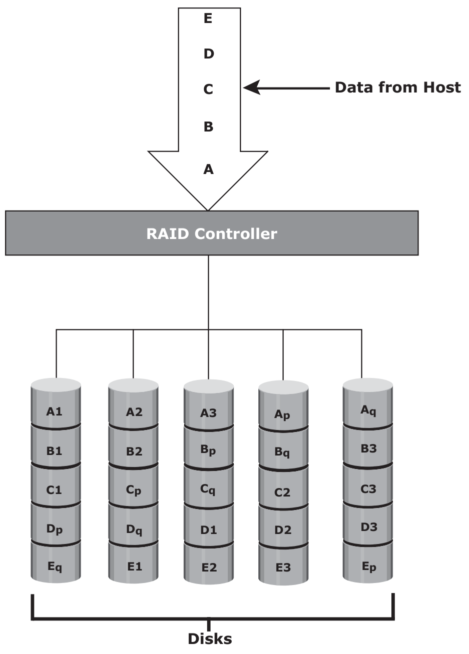

Figure 3-10: RAID 6

3.5 RAID Impact on Disk Performance

When choosing a RAID type, it is imperative to consider its impact on disk performance and application IOPS.

In both mirrored and parity RAID configurations, every write operation translates into more I/O overhead for the disks, which is referred to as a *write penalty*. In a RAID 1 implementation, every write operation must be performed on two disks configured as a mirrored pair, whereas in a RAID 5 implementation, a write operation may manifest as four I/O operations. When performing I/Os to a disk configured with RAID 5, the controller has to read, recalculate, and write a parity segment for every data write operation.

Figure 3-11 illustrates a single write operation on RAID 5 that contains a group of five disks.

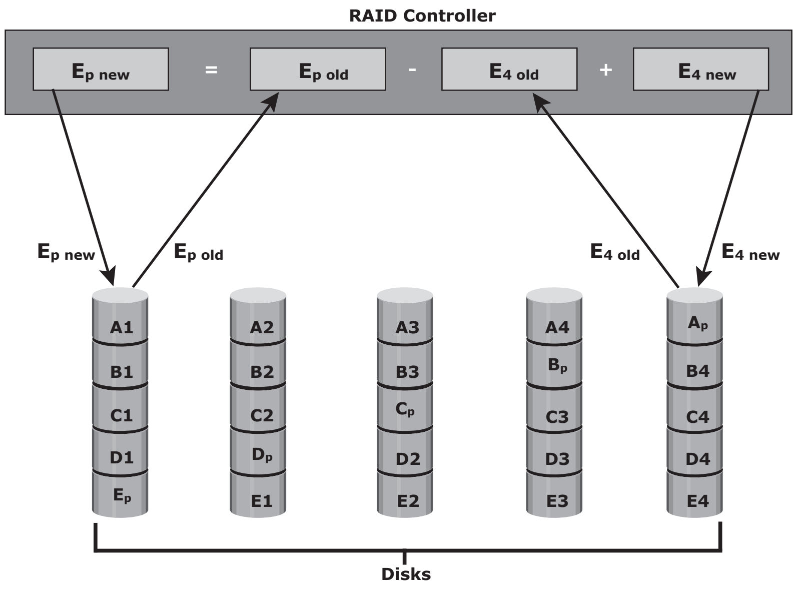

Figure 3-11: Write penalty in RAID 5

The parity (P) at the controller is calculated as follows:

$$E_p = E_1 + E_2 + E_3 + E_4 \text{ (XOR operations)}$$

Whenever the controller performs a write I/O, parity must be computed by reading the old parity (E_p old) and the old data ($E_{4\text{ old}}$) from the disk, which means two read I/Os. Then, the new parity (E_p new) is computed as follows:

$$E_{p\text{ new}} = E_{p\text{ old}} - E_{4\text{ old}} + E_{4\text{ new}} \text{ (XOR operations)}$$

After computing the new parity, the controller completes the write I/O by writing the new data and the new parity onto the disks, amounting to two write I/Os. Therefore, the controller performs two disk reads and two disk writes for every write operation, and the write penalty is 4.

In RAID 6, which maintains dual parity, a disk write requires three read operations: two parity and one data. After calculating both new parities, the controller performs three write operations: two parity and an I/O. Therefore, in a RAID 6 implementation, the controller performs six I/O operations for each write I/O, and the write penalty is 6.

3.5.1 Application IOPS and RAID Configurations

When deciding the number of disks required for an application, it is important to consider the impact of RAID based on IOPS generated by the application. The total disk load should be computed by considering the type of RAID configuration and the ratio of read compared to write from the host.

The following example illustrates the method to compute the disk load in different types of RAID.

Consider an application that generates 5,200 IOPS, with 60 percent of them being reads.

The disk load in RAID 5 is calculated as follows:

RAID 5 disk load (reads + writes) = 0.6 × 5,200 + **4** × (0.4 × 5,200) [because the write penalty for RAID 5 is 4]

= 3,120 + 4 × 2,080

= 3,120 + 8,320

= 11,440 IOPS

The disk load in RAID 1 is calculated as follows:

RAID 1 disk load = 0.6 × 5,200 + **2** × (0.4 × 5,200) [because every write manifests as two writes to the disks]

= 3,120 + 2 × 2,080

= 3,120 + 4,160

= 7,280 IOPS

The computed disk load determines the number of disks required for the application. If in this example a disk drive with a specification of a maximum 180 IOPS needs to be used, the number of disks required to meet the workload for the RAID configuration would be as follows:

RAID 5: 11,440/180 = 64 disks

RAID 1: 7,280/180 = 42 disks (approximated to the nearest even number)

3.6 RAID Comparison

Table 3-2 compares the common types of RAID levels.

Table 3-2: Comparison of Common RAID Types

RAID	MIN. DISKS	STORAGE EFFICIENCY %	COST	READ PERFORMANCE	WRITE PERFORMANCE	WRITE PENALTY	PROTECTION
0	2	100	Low	Good for both random and sequential reads	Good	No	No protection
1	2	50	High	Better than single disk	Slower than single disk because every write must be committed to all disks	Moderate	Mirror protection
3	3	[(n-1)/n] × 100 where n= number of disks	Moderate	Fair for random reads and good for sequential reads	Poor to fair for small random writes and fair for large, sequential writes	High	Parity protection for single disk failure
4	3	[(n-1)/n] × 100 where n= number of disks	Moderate	Good for random and sequential reads	Fair for random and sequential writes	High	Parity protection for single disk failure
5	3	[(n-1)/n] × 100 where n= number of disks	Moderate	Good for random and sequential reads	Fair for random and sequential writes	High	Parity protection for single disk failure
6	4	[(n-2)/n] × 100 where n= number of disks	Moderate but more than RAID 5.	Good for random and sequential reads	Poor to fair for random writes and fair for sequential writes	Very High	Parity protection for two disk failures
1+0 and 0+1	4	50	High	Good	Good	Moderate	Mirror protection

3.7 Hot Spares

A *hot spare* refers to a spare drive in a RAID array that temporarily replaces a failed disk drive by taking the identity of the failed disk drive. With the hot spare, one of the following methods of data recovery is performed depending on the RAID implementation:

- If parity RAID is used, the data is rebuilt onto the hot spare from the parity and the data on the surviving disk drives in the RAID set.
- If mirroring is used, the data from the surviving mirror is used to copy the data onto the hot spare.

When a new disk drive is added to the system, data from the hot spare is copied to it. The hot spare returns to its idle state, ready to replace the next failed drive. Alternatively, the hot spare replaces the failed disk drive permanently. This means that it is no longer a hot spare, and a new hot spare must be configured on the array.

A hot spare should be large enough to accommodate data from a failed drive. Some systems implement multiple hot spares to improve data availability.

A hot spare can be configured as automatic or user initiated, which specifies how it will be used in the event of disk failure. In an automatic configuration, when the recoverable error rates for a disk exceed a predetermined threshold, the disk subsystem tries to copy data from the failing disk to the hot spare automatically. If this task is completed before the damaged disk fails, the subsystem switches to the hot spare and marks the failing disk as unusable. Otherwise, it uses parity or the mirrored disk to recover the data. In the case of a user-initiated configuration, the administrator has control of the rebuild process. For example, the rebuild could occur overnight to prevent any degradation of system performance. However, the system is at risk of data loss if another disk failure occurs.

Summary

Individual disks are prone to failures and pose the threat of data unavailability. RAID addresses data availability requirements by using mirroring and parity techniques. RAID implementations with striping enhance I/O performance by spreading data across multiple disk drives, in addition to redundancy benefits.

This chapter explained the fundamental constructs of striping, mirroring, and parity, which form the basis for various RAID levels. Selection of a RAID level depends on the performance, cost, and data protection requirements of an application.

RAID is the cornerstone technology for several advancements in storage. The intelligent storage systems discussed in the next chapter implement RAID along with a specialized operating environment that offers high performance and availability.

EXERCISES

1. **Why is RAID 1 not a substitute for a backup?**
2. **Research RAID 6 and its second parity computation.**
3. **Explain the process of data recovery in case of a drive failure in RAID 5.**
4. **What are the benefits of using RAID 3 in a backup application?**
5. **Discuss the impact of random and sequential I/Os in different RAID configurations.**
6. **An application has 1,000 heavy users at a peak of 2 IOPS each and 2,000 typical users at a peak of 1 IOPS each. It is estimated that the application also experiences an overhead of 20 percent for other workloads. The read/write ratio for the application is 2:1. Calculate RAID corrected IOPS for RAID 1/0, RAID 5, and RAID 6.**
7. **For Question 6, compute the number of drives required to support the application in different RAID environments if 10 K RPM drives with a rating of 130 IOPS per drive were used.**
8. **What is the stripe size of a five-disk RAID 5 set with a strip size of 32 KB? Compare it with the stripe size of a five-disk RAID 0 array with the same strip size.**

Chapter 4
Intelligent Storage Systems

KEY CONCEPTS
Intelligent Storage Systems
Cache Mirroring and Vaulting
Logical Unit Number
LUN Masking
Meta LUN
Virtual Storage Provisioning
High-End Storage Systems
Midrange Storage Systems

Business-critical applications require high levels of performance, availability, security, and scalability. A disk drive is a core element of storage that governs the performance of any storage system. Some of the older disk-array technologies could not overcome performance constraints due to the limitations of disk drives and their mechanical components. RAID technology made an important contribution to enhancing storage performance and reliability, but disk drives, even with a RAID implementation, could not meet the performance requirements of today's applications.

With advancements in technology, a new breed of storage solutions, known as *intelligent storage systems*, has evolved. These intelligent storage systems are feature-rich RAID arrays that provide highly optimized I/O processing capabilities. These storage systems are configured with a large amount of memory (called *cache*) and multiple I/O paths and use sophisticated algorithms to meet the requirements of performance-sensitive applications. These arrays have an operating environment that intelligently and optimally handles the management, allocation, and utilization of storage resources. Support for flash drives and other modern-day technologies, such as virtual storage provisioning and automated storage tiering, has added a new dimension to storage system performance, scalability, and availability.

This chapter covers components of intelligent storage systems along with storage provisioning to applications.