An Introduction to RAID
Redundant Arrays of Inexpensive Disks

Pete McLean

24 April 1991

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

Copyright © 1991 by Digital Equipment Corporation

All Rights Reserved.

Digital Equipment Corporation

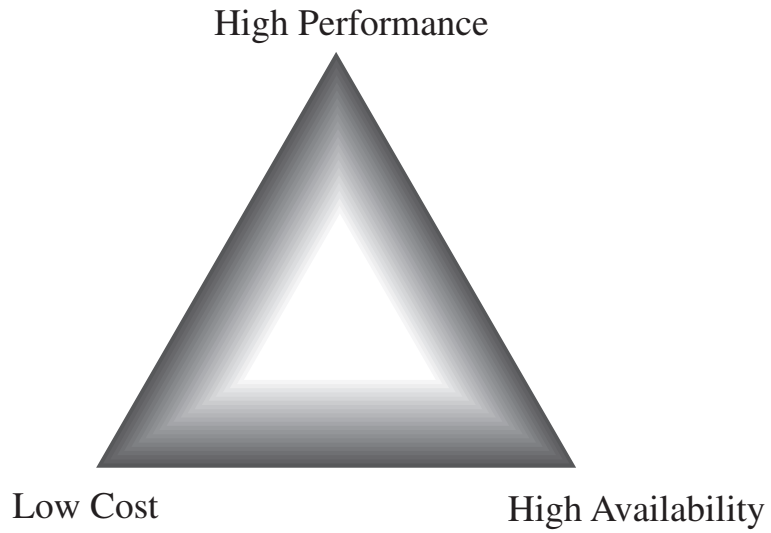
CHAPTER 1

INTRODUCTION

This paper is written to provide an introduction for those uninitiated in the concept of RAID (Redundant Arrays of Inexpensive Disks). It is not a deep technical presentation, there are a number of supporting documents that treat the subject in greater depth. This paper is intended to provide the reader with a brief history of the problems leading to RAID, the definition of RAID, a description of the various RAID levels (configurations) being discussed in the industry today, and a comparison of those RAID levels.

It is hoped that having read this paper, the reader will have a fundamental understanding of what RAID is, as well as an understanding of when and why RAID is beneficial. The requirements of all systems users fall into the triangle shown in Figure 1. For some users, a system has three important attributes. Low cost, low cost, and low cost. These users occupy the lower left hand corner of the triangle. Other users may lose millions of dollars in business if their system is down for only a short time. For them, additional dollars spent to provide high availability are dollars well spent. They occupy the lower right hand corner of the triangle. For still others, getting the job done quickly outweighs any other attribute. They occupy the top of the triangle.

Figure 1: User Requirements



Most users in fact fall somewhere more in the middle of the triangle. While placing dollar values on each attribute may not be easy, most can at least rank the importance of the attributes and place themselves in a general area.

This paper attempts to provide the reader with guidance as to what RAID solution may be of interest based on their particular system needs.

CHAPTER 2

SOME HISTORY

Over the past two decades, the processing power of CPU's has been growing at a rate of five to ten fold every three years depending on the source you choose to believe. During that same period, disk storage has doubled in capacity and halved in cost every three years, but due to its electromechanical nature, seek and rotational latency has only improved 10 to 50 percent in the last decade. These trends have caused many to predict a dire I/O crisis is upon us.

With no new technology on the horizon to replace magnetic or optical disks as secondary storage, and with the laws of mechanics limiting what can be done to improve these products, other avenues were investigated to help solve the I/O problem.

The nature of the user's I/O workload was investigated. The studies concluded that in fact the user's workload was very poorly distributed across the storage. So called "hot spots", areas of very high I/O access, were common while other areas had little or no activity. In some cases these "hot spots" could be attributed to a single file with a great deal of activity. In other cases, it would be a number of these "hot" files that happened to be located on the same disk drive. From these observations came the so called "80/20 Rule". That is, it was noted that on most systems, about twenty percent of the storage capacity received about eighty percent of the I/O requests issued by the user. If this twenty percent of the capacity was located on a single drive, the performance was poor. If it were spread across a number of drives, performance could be improved.

With the introduction of the RA70, Digital began to tell customers to place their "hot files" on a set of new RA70's, leaving their less frequently accessed files on larger RA82 or RA90 drives. This provided more actuators to serve the "hot files" and at the same time provided lower cost storage for the remainder of the files. For those who could identify a static set of "hot files", this worked well. However many customers either could not identify the "hot files" or had systems in which the files that were "hot" would vary with time of day or week of month, etc.

To help solve this problem in a more general way, the concept of striping was introduced. With striping, the user views a set of physical disk drives as a single logical disk drive and the data is broken into chunks and distributed over the set. The details of this will be described in detail later, but the effect is to distribute the "hot spots" across the set of drives. With this distribution, each drive tends to get a more equal share of the I/O request load and the aggregate performance improves.

CHAPTER 3

THE BERKELEY RAID PAPER

In December, 1987, a paper entitled "A Case for Redundant Arrays of Inexpensive Disks (RAID)" was published by David Patterson, Garth Gibson, and Randy Katz of the University of California, Berkeley. They included some of the industry trends described here and noted that distributing the data across more, smaller drives could improve performance. In addition, they pointed out that as more and more smaller drives were used to gain performance, the MTBF of the set of drives went down. That is, smaller drives each have an MTBF that is negligibly better than larger drives. As the number of drives used increased, the aggregate MTBF of the set dropped dramatically. Performance would go up but availability would suffer.

To solve this MTBF problem, the paper introduced the concept of RAID (Redundant Arrays of Inexpensive Disks) and described five possible RAID levels or configurations.

As defined, RAID has three attributes.

1. It is a set of physical disk drives viewed by the user as a single logical device.
2. The user's data is distributed across the physical set of drives in a defined manner.
3. Redundant disk capacity is added so that the user's data can be recovered even if a drive fails.

The manner in which the data is distributed and the manner in which the redundant capacity is implemented are the distinguishing features among the various RAID levels.

Of the five RAID levels described in the paper, RAID-1, RAID-3, and RAID-5 have met with significant interest in the industry and will be described here. In addition, two further RAID levels have been added by the industry, RAID-0 and RAID-6, and will also be described. RAID-2 and RAID-4 will be touched on briefly in the RAID-3 and RAID-5 chapters.

CHAPTER 4

RAID-0

RAID-0 was not described in the Berkeley paper. Its definition has been created by the industry. It is in fact, striping as referred to previously and as offered by Digital as our VMS layered striping product.

RAID-0 has no redundancy and is really a misnomer. It probably should be called AID-0 with no R. As a result, any disk failure will cause the lack of ability to access the data and may result in permanent loss.

Referring to Figure 2, the RAID set is viewed by the user as a single logical device containing a specified storage capacity. The figure shows a set of four physical disks in the RAID. This number is arbitrary and in fact is chosen here simply to demonstrate the configuration within the confines of a page.

The user's logical view of the data is broken up into segments referred to as "chunks", each chunk being noted with a letter, i.e., A, B, etc. The first chunk, A, is physically located on the first disk. The second chunk, B, is placed on the second disk at the same physical location as A was placed on the first disk. The third chunk, C, is placed on the third disk and the fourth chunk, D, is placed on the fourth disk. The fifth chunk, E, is then placed on the first disk following chunk A; the sixth, F, on the second disk following chunk B, and so on until the entire logical device capacity has been distributed across the set of physical disks.

In this example, the size of the chunk used to distribute the data across the set is assumed large with respect to a user request. A user request as shown above chunk B is therefore a small portion of chunk B as shown by the shaded area. When that request is read or written, only the shaded area of chunk B on the second drive is accessed. With this request size to chunk size ratio, almost all requests will fall within a single chunk.

To read the array, the user request is mapped to the physical location where the data resides on the physical disk. The data is then read from that location. Similarly, for a write, the location is mapped and the data written to that physical location. Since almost all requests will fall within a single chunk, each of the disks in the set will be able to execute a different request, providing the performance of all of the disk actuators simultaneously. This provides good performance for a user workload that is a high request rate of small to medium sized requests.

Since no redundancy exists, no recovery is possible in the event of a disk failure. Access is lost if any one of the drives fails.

In terms of cost of storage, the user need only buy the capacity needed to store his data.

Figure 2: RAID-0 - Small Requests

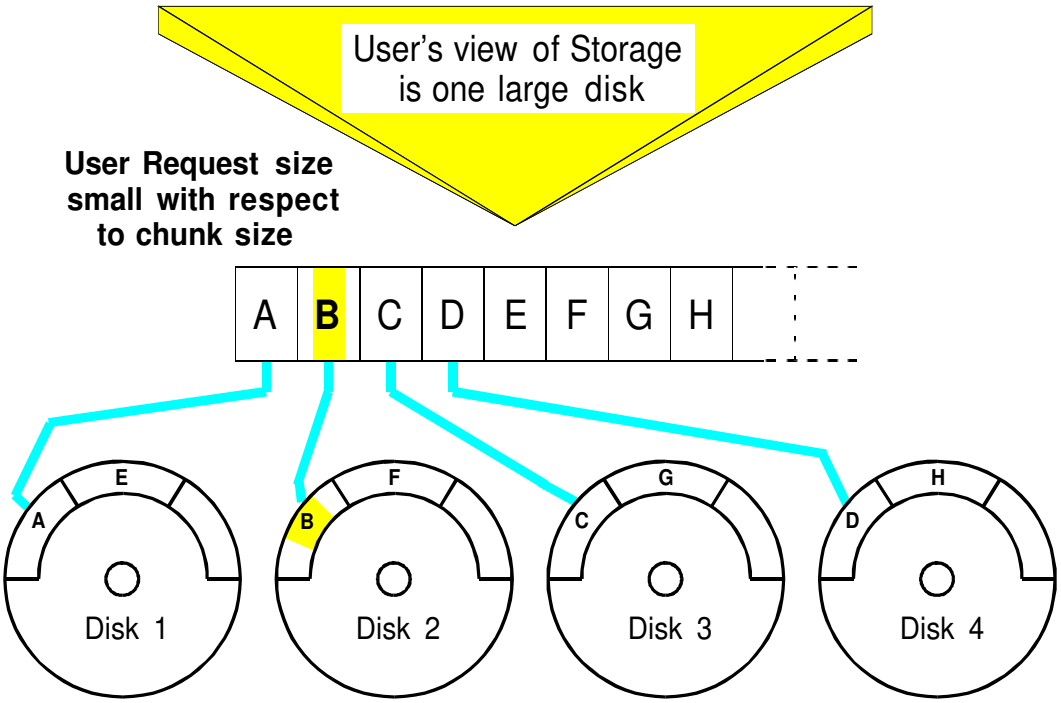
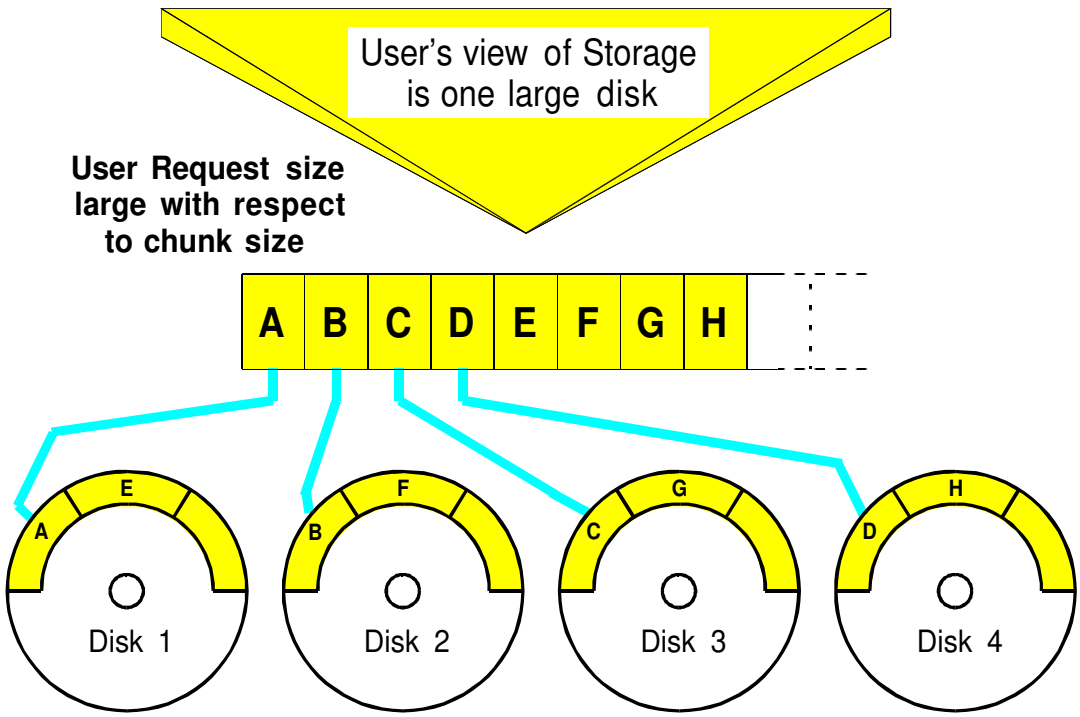


Figure 3 shows a slightly different picture of RAID-0. Again, the user views the storage as a single large device and the data is distributed across the set of disks in chunks. However, in this case the requests are assumed to be large and the chunk size is chosen to be small with respect to the request size. As can be seen from the shaded area, the single request is distributed across all of the disks. This means that all of the actuators will move together to the same location to serve each request and each drive will transfer a portion of the request simultaneously. This provides a very high transfer rate for each large request but limits the number of requests serviced since only one request can transfer at a time. With this form of striping, RAID-0 provides very good performance for the user whose workload is a near sequential stream of large requests.

In terms of cost and redundancy, the configuration yields the same characteristics as previously described for RAID-0.

To summarize RAID-0 qualitatively, it is the best RAID in terms of cost. The customer need not pay for extra capacity for redundancy. The data integrity is poor however, since no redundancy is provided and a single failure will cause loss of access and perhaps loss of data. The performance is good. The workload is balanced across the disks, the chunk size can be chosen to provide the best possible performance for the workload, and there is no extra overhead for writing redundant information.

Figure 3: RAID-0 - Large Requests



CHAPTER 5

RAID-1

RAID-1 was the first RAID level described in the Berkeley paper. It is in fact shadowing as Digital has been offering since the mid 80's.

Referring to Figure 4, the RAID set is viewed by the user as a single logical device containing a specified storage capacity.

The user's logical view of the data is broken up into segments referred to as "chunks", each chunk being noted with a letter, i.e., A, B, etc. The data is placed as has been done on traditional disk configurations. That is, chunks A, B, C, ... are placed in order on the first disk. When the first disk is filled, this procedure would continue on the third disk and so on. As can be seen, this distribution does nothing to improve the "hot spot" problem. The set of disks will exhibit the 80/20 problem.

Redundancy is provided by a second disk which is an exact copy of the first. That is, chunks A, B, C, ... are placed on the second disk as well as the first disk in exactly the same disk locations. Likewise, the fourth disk would be an exact copy of the third.

To read the array, the user request is mapped to the physical location where the data resides on the physical disks. The data can then be read from either of the two disks where it resides so that each of the shadowed pair of disks executes only half of the read requests. For a write, the location is mapped and the data is written to both members of the pair.

If a disk fails, there is no interruption of the user's access to his data or loss of his data. All reads and writes are simply executed by the remaining good disk of the pair.

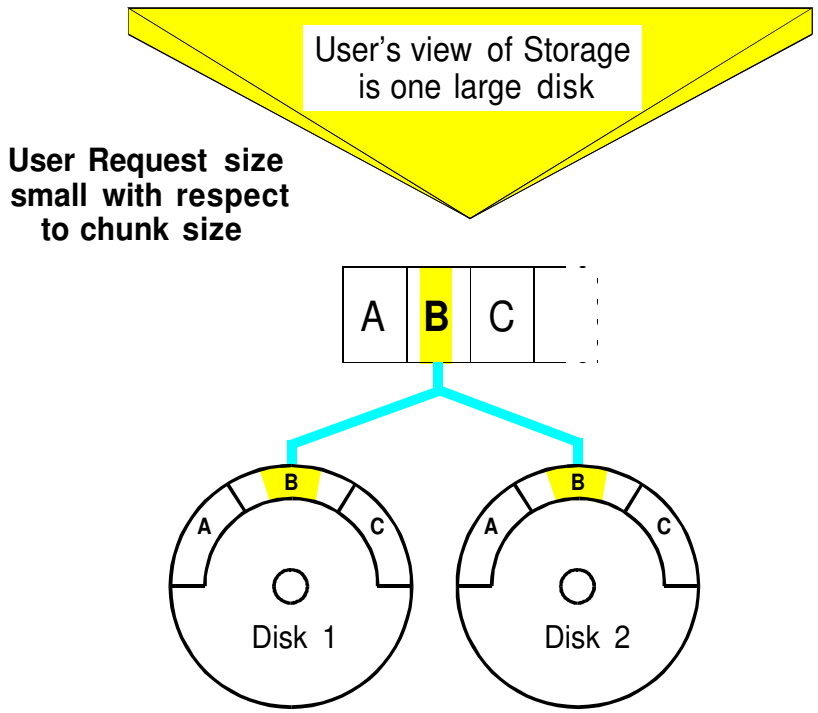
The availability of this configuration is excellent. If one disk fails, there is no loss of data availability unless the second disk of the pair also fails before the first can be replaced.

In terms of cost of storage, however, it is expensive. The user must purchase twice the data capacity he actually uses.

Figure 5 shows a slightly different picture of RAID-1. Again, the user views the storage as a single large device. However, in this case the user's data is striped across the drive set as was shown in the first case for RAID-0.

The user's logical view of the data is broken up into segments referred to as "chunks", each chunk being noted with a letter, i.e., A, B, etc. The first chunk, A, is physically located on the first and second disk. The second chunk, B, is placed on the third and fourth disk at the same physical location as A was placed on the first disk. The third chunk, C, is then placed on the first and second disk following chunk A; the fourth D, on the third and fourth disk following chunk B, and so on until the entire logical device capacity has been distributed across the set of physical disks.

Figure 4: RAID-1



In this example, the size of the chunk used to distribute the data across the set is assumed large with respect to a user request. A user request as shown above chunk B is therefore a small portion of chunk B as shown by the shaded area. When that request is read, only the shaded area of chunk B on the third or the fourth drive is accessed. When the request is a write, the shaded areas on both the third and fourth drives are written. With this request size to chunk size ratio, almost all requests will fall within a single chunk. Each of the disks in the set can execute a different read request simultaneously. Each shadowed pair can execute a write simultaneously. This provides good performance for a user workload that is a high request rate of small to medium sized requests.

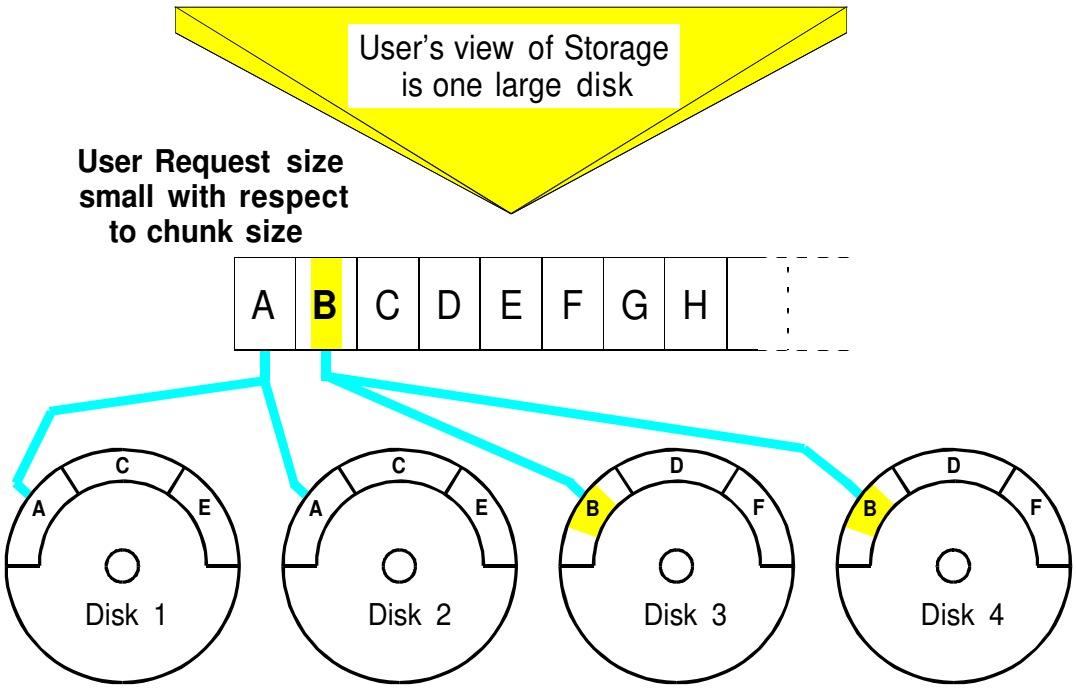
In terms of cost and redundancy, the configuration yields the same characteristics as previously described for RAID-1.

One more example could be shown but will be omitted. It should be easy to visualize that if the user requests were large and the chunk size were chosen small with respect to it, the same high performance that is gained in RAID-0 for near sequential large requests could also be gained with RAID-1.

To summarize RAID-1 qualitatively, it is the highest cost RAID. The customer must pay for twice the disk capacity he needs. The data integrity, however, is very good. The user does not lose data unless the second of a pair fails before the first is replaced. When striped, the performance is the best of all RAID configurations. The workload is balanced across the disks, the chunk size can be chosen to provide the best possible performance for the user's

workload. Each disk need execute only half of the read request load and the write to each of the pair can be done simultaneously.

Figure 5: RAID-1 - Striped



CHAPTER 6

RAID-3

RAID-3 was also described in the Berkeley paper.

Referring to Figure 6, the RAID set is viewed by the user as a single logical device containing a specified storage capacity. The figure shows a set of four physical disks in the RAID. This number is arbitrary.

The user's logical view of the data is broken up into segments referred to as "chunks", each chunk being noted with a letter, i.e., A, B, etc. The first chunk, A, is physically located on the first disk. The second chunk, B, is placed on the second disk at the same physical location as A was placed on the first disk. The third chunk, C, is placed on the third disk. The fourth chunk, D, is then placed on the first disk following chunk A; the fifth, E, on the second disk following chunk B, and so on until the entire logical device capacity has been distributed.

The user's requests are assumed to be large and the chunk size is chosen to be small with respect to the request size. The typical chunk size in RAID-3 is one bit or one byte. As can be seen from the shaded area, the single request is distributed across all of the disks. This means that all of the actuators will move together to the same locations to serve each request and each disk will transfer a portion of the request simultaneously. This provides a very high transfer rate for each large request but limits the number of requests serviced since only one request can transfer at a time. This provides very good performance for the user whose workload is a near sequential stream of large requests.

With RAID-3, the redundancy is provided by the fourth disk shown. The first chunk on the fourth disk contains the Exclusive Or (Xor) of the first chunks on the three data disks, i.e., A $\oplus$ B $\oplus$ C. The second chunk on the fourth disk, the Xor of the second chunks on the data disks, i.e., D $\oplus$ E $\oplus$ F, and so on.

To read the array, the user request is mapped to the physical location where the data resides on the physical disks. The data is then read simultaneously from all of the data disks. Similarly, for a write, the location is mapped and the data written is simultaneously to all of the data disks. As the write is executing, the Xor is calculated for the data as it passes and the Xor is also written simultaneously. If a drive fails, the data from the failed drive can be reconstructed as a read is executed by reading the Xor disk as well as the remaining data disks and Xoring all the chunks together to yield the missing data.

RAID-3 is somewhat different from RAID-1 in that the chunk size is intimately tied to the redundancy mechanism. That is, because it is assumed that all of the data required to calculate the Xor is available when a write is executed or a read is done with a failed drive, the chunk size must be small with respect to the request size so that all of the chunks involved in the Xor are transferred by the request.

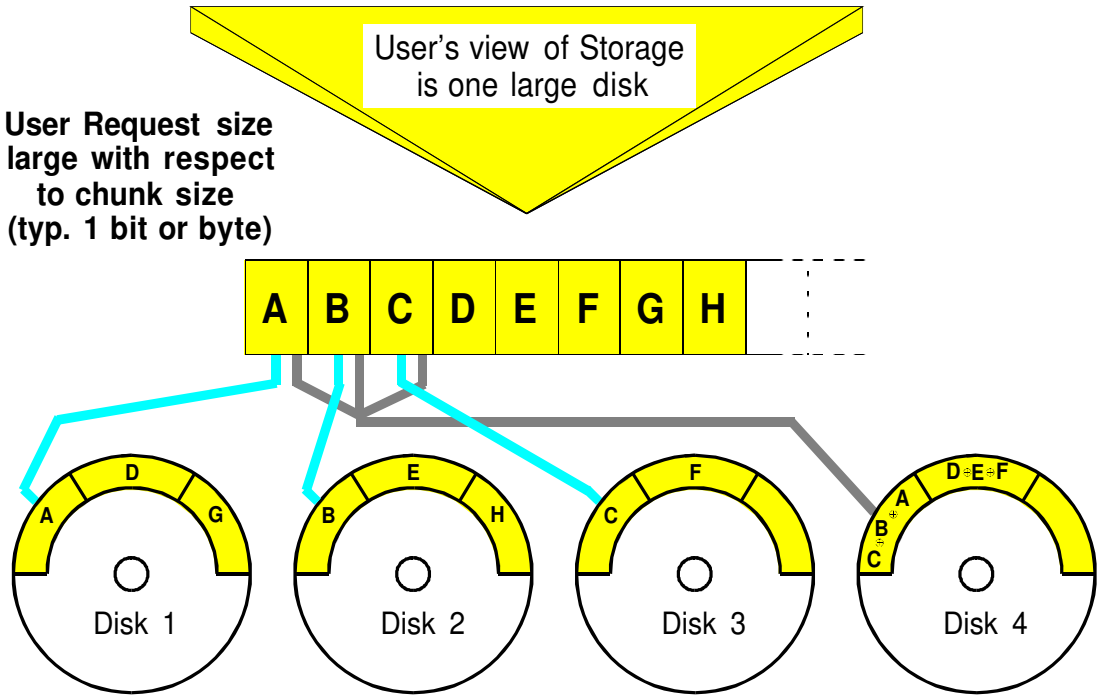
RAID-3 provides good data integrity since a second member of the Xor set must fail before a first failure is replaced for the user to lose access to his data or possibly lose his data.

In terms of cost of storage, RAID-3 is also good. The user need buy just one additional disk to provide the redundancy.

To summarize RAID-3 qualitatively, it is good in terms of cost. The customer needs just one extra disk for redundancy. The data integrity is also good. A second member of the array must fail before a first failure is replaced to lose access or perhaps data. The performance is good if the workload is a near sequential stream of large requests. The configuration really looks like a single disk with a very high transfer rate.

RAID-2 is very similar to RAID-3 except it has a number of redundant disks forming a Hamming code rather than the single Xor disk. Thus, the redundancy is more costly.

Figure 6: RAID-3



CHAPTER 7

RAID-5

RAID-5 was also described in the Berkeley paper.

Referring to Figure 7, the RAID set is viewed by the user as a single logical device containing a specified storage capacity. The figure shows a set of four physical disks in the RAID, this number is arbitrary.

The user's logical view of the data is broken up into segments referred to as "chunks", each chunk being noted with a letter, i.e., A, B, etc. Redundancy is provided by an Xor similar to RAID-3, but in the case of RAID-5, the Xor chunks are also distributed across the drives. Therefore, the first chunk of user data, A, is physically located on the first disk. The second chunk, B, is placed on the second disk at the same physical location as A was placed on the first disk. The third chunk, C, is placed on the third disk. The first chunk location on the fourth disk contains A $\oplus$ B $\oplus$ C. The fourth chunk of user data, D, is placed on the first disk following chunk A; the fifth, E, on the second disk following chunk B. The second chunk location on the third disk following chunk C, contains D $\oplus$ E $\oplus$ F, and the sixth user chunk, F, is placed on the fourth disk. For the next set of user chunks, G, H, and I, the Xor is placed on the second disk. This pattern is repeated until the entire user data space has been distributed across the disks.

The size of the chunk used to distribute the data across the set is large with respect to a user request. A user request as shown above chunk B is therefore a small portion of chunk B as shown by the shaded area. When that request is read, only the shaded area of chunk B on the second drive is accessed. When that request is written, only the shaded area of chunk B on the second disk and its associated Xor, the shaded area on the fourth disk are accessed. With this request size to chunk size ratio, almost all requests will fall within a single chunk.

To read the array, the user request is mapped to the physical location where it resides on the physical disk. The data is then read from the data disk. For a write, the location is mapped and then the old data in that location and its associated Xor data are read. The old data is then Xored with the old Xor data thus removing the old data from the old Xor data. The new data to be written is then Xored with the result from the previous Xor providing the new Xor data. The new data and the new Xor data are then written to the two disks involved. Hence, a write request for RAID-5 involves two disks and each has an extra revolution of latency. Or, from a different point of view, a write requires four transfers: a read followed by a write for the data and a read followed by a write for the Xor. In the event of a disk failure, all of the remaining disks are read, and the chunks from each Xored together to reconstruct the missing data. Since almost all requests will fall within a single chunk, each of the disks in the set will be able to execute a different read request simultaneously. Writes will involve two disks but the remaining disks are free to execute other requests. Since the Xor chunks are distributed across the set of disks, it is possible to do simultaneous writes.

This provides good performance for a user workload that is a high request rate of small to medium sized requests.

With RAID-5 the chunk size is also intimately tied to the redundancy mechanism. That is, because it is assumed that only a portion of the data required to calculate the Xor is available when a write is executed, the read followed by write algorithm is used.

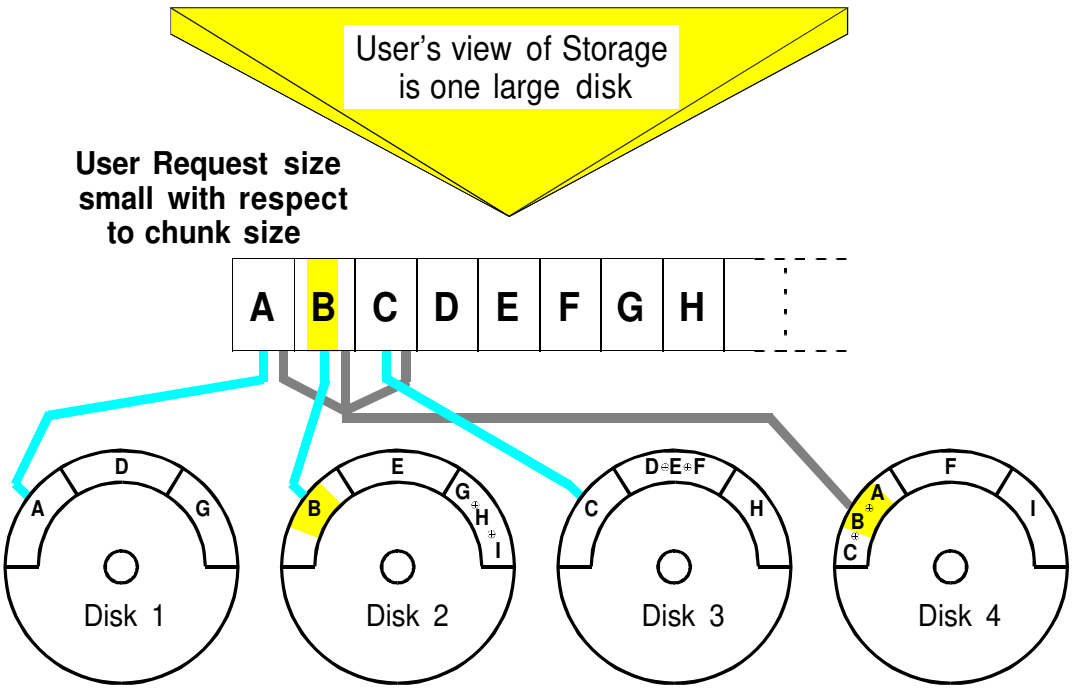
RAID-5 provides good data integrity since a second member of the Xor set must fail before a first failure is replaced for for the user to lose access to his data or to possibly lose his data.

In terms of cost of storage, RAID-5 is also good. The user need buy just one additional disk to provide the redundancy.

To summarize RAID-5 qualitatively, it is good in terms of cost. The customer needs just one extra disk for redundancy. The data integrity is also good. A second member of the array must fail before a first failure is replaced to lose access or perhaps data. The performance is moderate. Hot spots are distributed by the distribution of data, but the extra latency of write requests has a detrimental effect on overall performance.

RAID-4 is exactly like RAID-5 except that all of the Xor chunks are located on one disk as in RAID-3. As a result, all writes must access a single disk and only one write may be executed at a time.

Figure 7: RAID-5



CHAPTER 8

RAID-6

RAID-6 was was not described in the Berkeley paper but has been introduced by the industry since.

Referring to Figure 8, the RAID set is viewed by the user as a single logical device containing a specified storage capacity. The figure shows a set of four physical disks in the RAID. This number is arbitrary.

The user's logical view of the data is broken up into segments referred to as "chunks", each chunk being noted with a letter, i.e., A, B, etc. With RAID-6, the redundancy is provided by redundancy chunks on two disks for each set of data chunks. The redundancy is calculated using a Reed-Solomon code creating redundancy chunks called P and Q. One can think of this as calculating two results for the same input using two independent equations. If two drives fail, one is left with two equations and two unknowns and the missing data can be recreated. The P and Q chunks are distributed across the set as RAID-5 does with its Xor. Therefore, the first chunk of user data, A, is physically located on the first disk. The second chunk, B, is placed on the second disk at the same physical location as A was placed on the first disk. The first chunk location on the third disk contains P(A,B), and fourth disk contains Q(A,B). The third chunk of user data, C, is placed on the first disk following chunk A. The second chunk location on the second disk contains P(C,D) and third disk contains Q(C,D). The fourth chunk of user data, D, is placed on the fourth disk. The third chunk location on the first disk, following C, contains P(E,F); the third chunk location on the second disk, Q(E,F). User data chunk E, is then placed on the third disk, and F on the fourth. This pattern is repeated until the entire user data space has been distributed across the disks.

The size of the chunk used to distribute the data across the set is large with respect to a user request. A user request as shown above chunk B is therefore a small portion of chunk B as shown by the shaded area. When that request is read, only the shaded area of chunk B on the second drive is accessed. When that request is written, only the shaded area of chunk B on the second disk and its associated P and Q, the shaded areas on the third and fourth disks, are written. With this request size to chunk size ratio, almost all requests will fall within a single chunk.

To read the array, the user request is mapped to the physical location where it resides on the physical disk. The data is then read from the data disk. For a write, the location is mapped and then the old data in that location and its associated P and Q are read. The old data is removed from the old P and Q. The new data to be written is then used to calculate the new P and Q. The new data and the new P and Q are then written to the three disks involved. Hence, a write request for RAID-6 involves three disks and each has an extra revolution of latency. Or, from a different point of view, a write requires six transfers: the read of the data followed the write of the data, the read of the P followed by the write of the P, and the read of the Q followed by the write of the Q. In the event of a disk failure, all of the remaining data chunks are read, plus P or Q, and the missing data is computed.

In the event of two disk failures, all of the remaining disks are read and the missing data computed. Since almost all requests will fall within a single chunk, each of the disks in the set will be able to execute a different read request simultaneously. A write request will involve three disks but the remaining disks can execute other requests. Since the P and Q are distributed across the set of disks, simultaneous writes can be executed if the set of disks is large enough. This provides good performance for a user workload that is a high request rate of small to medium sized requests.

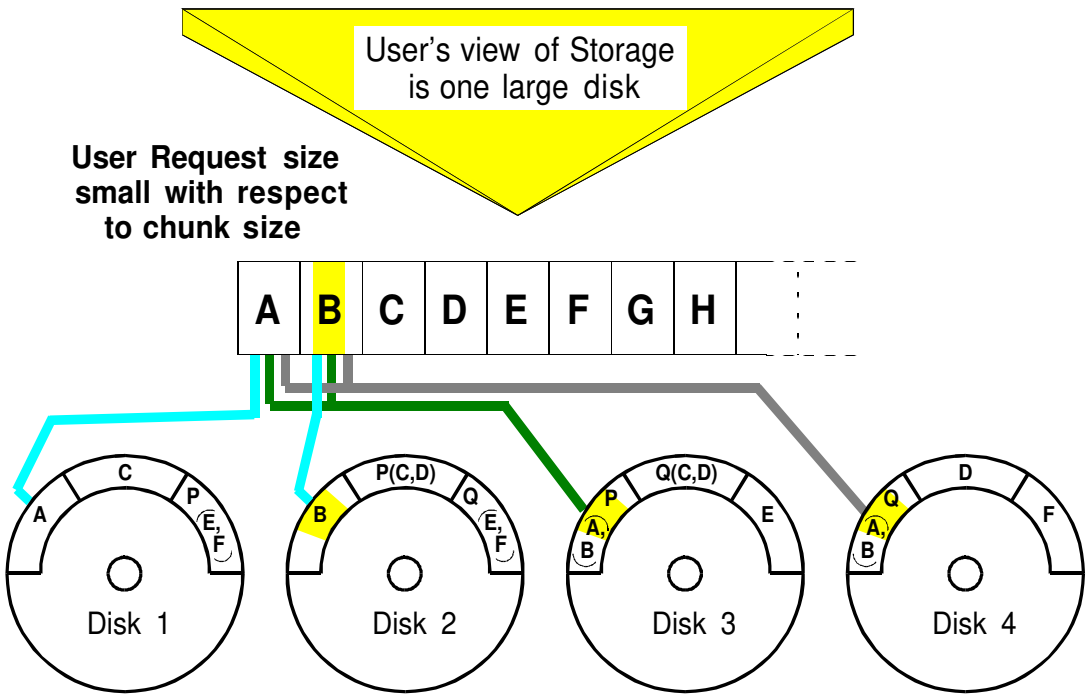
With RAID-6 the chunk size is also intimately tied to the redundancy mechanism. That is, because it is assumed that only a portion of the data required to calculate the P and Q is available when a write is executed, the read followed by write algorithm is used.

RAID-6 provides very good data integrity since two additional members of the set must fail before a first failure is replaced for the user to lose access to his data or to possibly lose his data.

In terms of cost of storage, RAID-6 is moderate. The user must buy two additional disks to provide the redundancy.

To summarize RAID-6 qualitatively, it is moderate in terms of cost. The customer buys two extra disks for redundancy. The data integrity is the best. Two members of the array must fail before a first failure is replaced to lose access or perhaps data. The performance is moderate. Hot spots are distributed by the distribution of data, but the extra latency of write requests has a detrimental effect on overall performance.

Figure 8: RAID-6



CHAPTER 9

RAID IMPLEMENTATIONS

The descriptions of the various RAID levels has shown tradeoffs among them in terms of cost, availability, and performance. It should be recognized however that the manner in which they are implemented will also affect these attributes.

For example, a RAID could be implemented with very inexpensive disk drives yielding a product that would be very cost effective. However, these very inexpensive drives may not have the best MTBF or performance available, so the implementation may sacrifice those attributes. On the other hand, an implementation with the highest performing drives possible or best MTBF possible, may be substantially more expensive.

In general, higher capacity drives have a lower cost per megabyte. A RAID implementation with these drives will yield a lower cost per megabyte product but performance will be sacrificed since the number of actuators per megabyte of capacity will be lower.

In the description of the various RAID levels, the RAID functions, i.e., mapping the user view to the physical disks, calculating the redundancy, and recovering data in the face of a failure, were discussed. These functions must be performed by some entity between the user and the set of disks. This entity could be implemented in a very high performance manner, resulting in high product cost, or it could be implemented in a very cost effective manner, sacrificing some performance.

Finally, it should be recognized that RAID really addresses loss of user data at the disk level. If the RAID function is implemented as a single function, which can fail and cause loss of access to the user's data by doing so, the implementation does not have high availability. If the failure of the power supplies serving the disks or the RAID function can fail and cause loss of access, the implementation does not have high availability. The same is true if the interconnect to the system is a single point of failure. Therefore, the redundancy provided by the implementation in terms of the RAID function, power, and interconnects will have a larger impact on the availability of the product than RAID itself.

CHAPTER 10

A RAID EXAMPLE

Now that the RAID levels have been described, let's compare them with an example. Please recognize that this example compares only the fundamental attributes of the levels. It compares the same user capacity with the same disk drives. It does not include the additional effects to cost, performance, and availability due to the implementation as described in the previous section.

For the example, the RAIDs use a 5 1/4" disk drive typically available in the market today. Each RAID level is configured to provide four drives of user capacity. The MTBF for each configuration is calculated based on a 150K hour drive MTBF and a 24 hour Mean Time to Repair. This 24 hour figure is not unreasonable when one remembers that repair of the RAID is not completed until the failed drive is replaced and all of the data that belongs on it has been recreated and written to it.

Since the different RAID levels have different performance depending on the user workload, the performance is compared for both small and large request sizes. In both cases, a workload consisting of 25% writes and 75% reads is used with a typical VMS locality of reference.

10.1 Example Cost

Figure 9 shows a cost comparison for the required storage capacity. Quite simply, RAID-0 which was qualitatively described as best requires just the cost of four disks. RAID-3 and RAID-5 require one extra disk of redundancy, thus a 25% increase in cost. RAID-6 requires two disks of redundancy, a 50% increase. Finally, RAID-1 is most expensive, with a cost that is twice that of RAID-0.

10.2 Example Availability

The MTBF for each level is shown in Table 1. Please note that these results are expressed in M hours, million hours. A million hours is about 114 years. The numbers vary by so many orders of magnitude that it is impossible to reasonably show them on a graph. As can be seen, RAID-6 is best, RAID-1 is very good, RAID-3 and RAID-5 are good, and RAID-0 is poor.

Figure 9: Example Cost

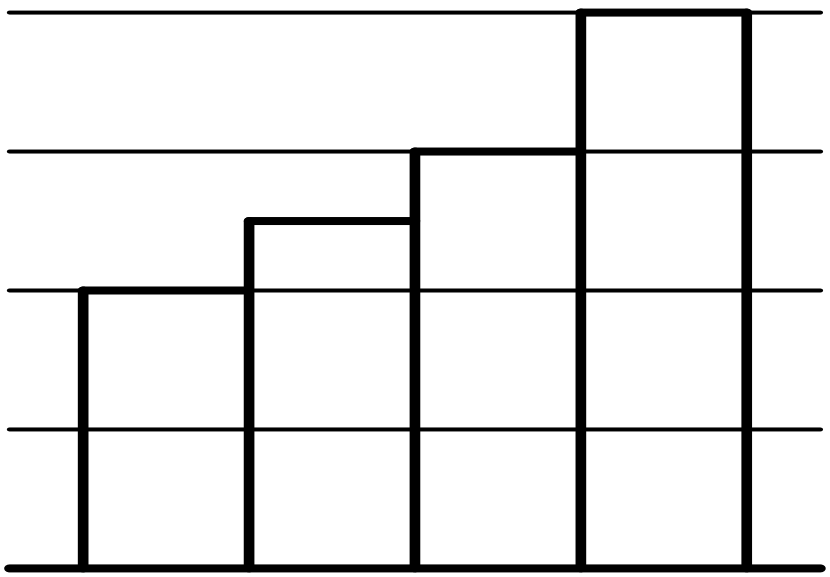


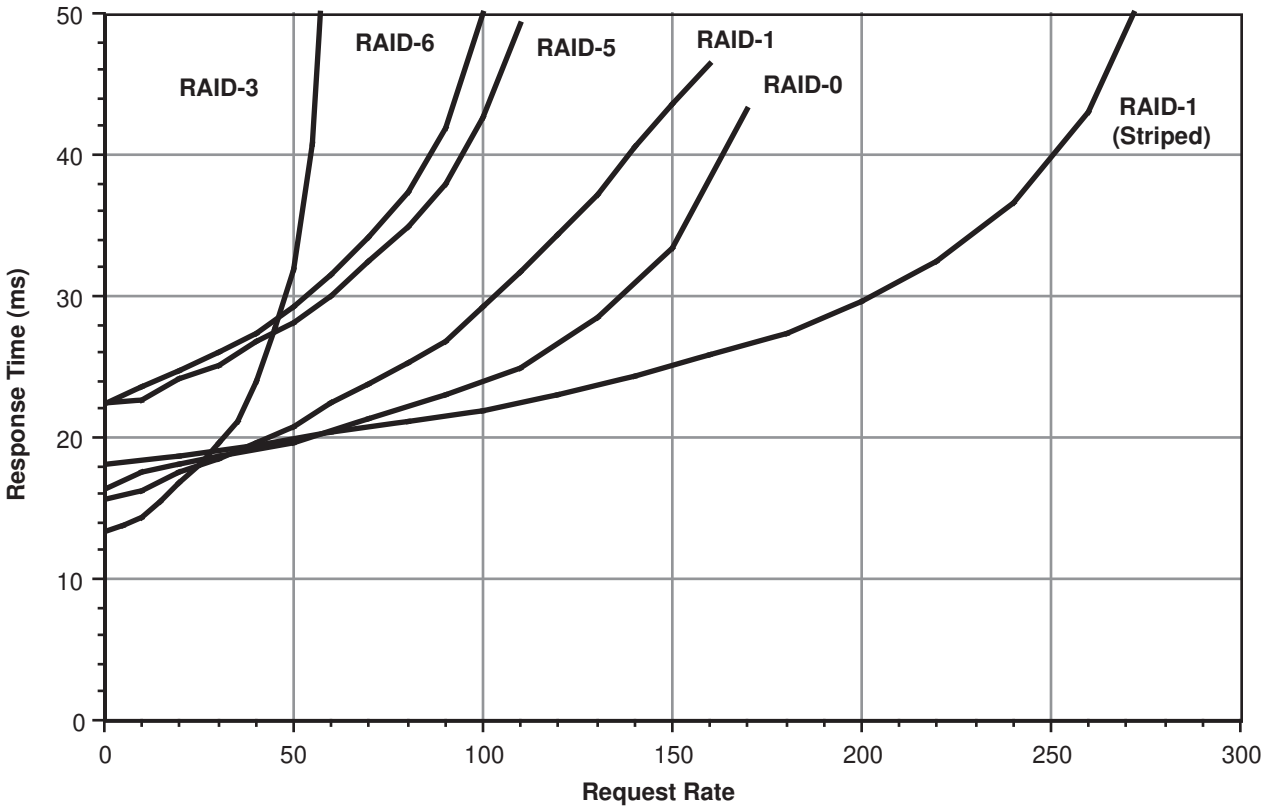
Table 1: Example MTBF

RAID Level	MTBF
RAID-0	0.0375 M *hours*
RAID-1	117.0 M *hours*
RAID-3	47.0 M *hours*
RAID-5	47.0 M *hours*
RAID-6	6000.0 M *hours*

10.3 Example Performance

Figure 10 shows the mean response time vs the I/O rate in requests per second for each RAID. These curves represent the performance for small requests, four sectors, with 25% writes. For the RAID-0, RAID-1 striped, RAID-5, and RAID-6 configurations, the chunk size is six tracks so that very few requests involve more than one chunk.
From these curves it can be seen that RAID-0 is good. It distributes the "hot spots" and has no extra write overhead. RAID-1 without striping isn't bad but it suffers from the 80/20 problem. RAID-1 striped is best. It distributes the load, reads are distributed between each member of the shadow pair, and the pairs of writes can be done simultaneously. RAID-3 behaves like a single actuator. It is not at all suited for this workload. RAID-5 and RAID-6 have moderate performance. They do distribute the load but the penalty for the extra revolution on writes reduces performance.

Figure 10: Example Performance - Small Requests

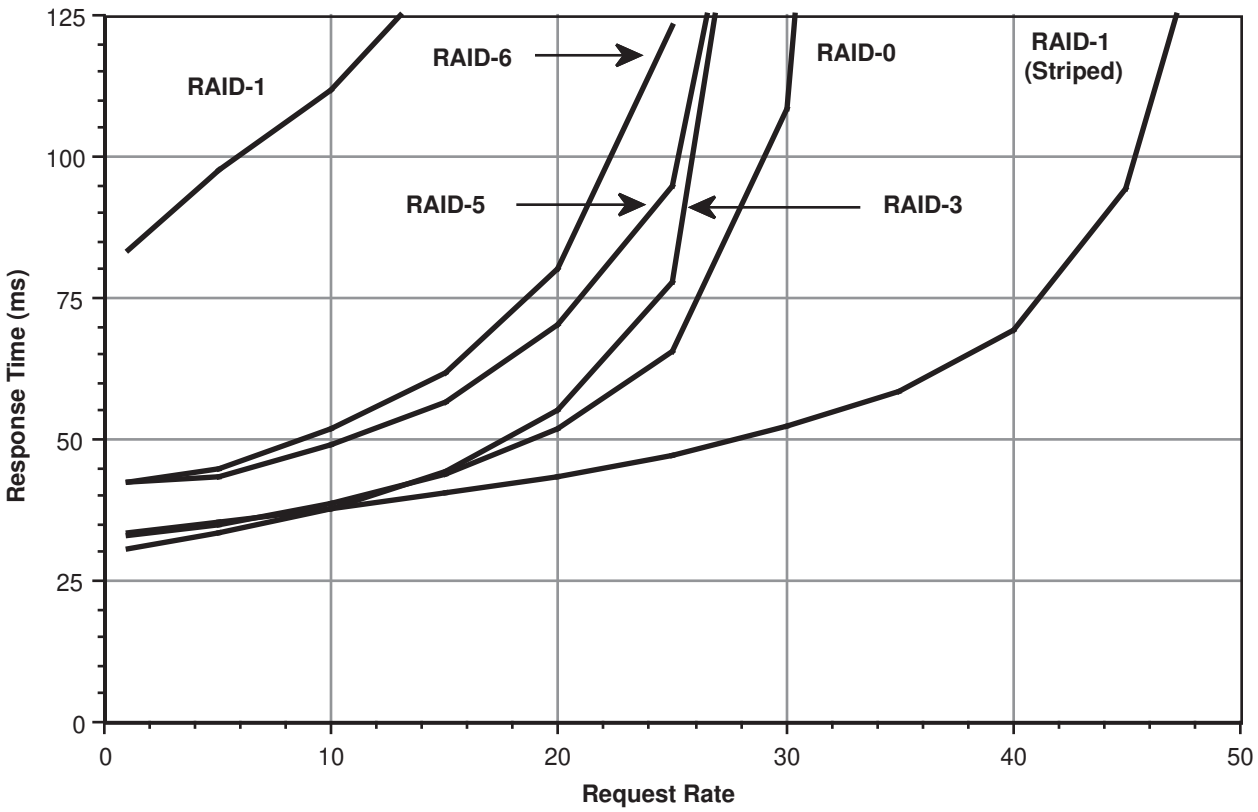


It is interesting to note the comparison between the RAID-1 striped and unstriped curves. Unstriped, the 80/20 effect dominates. Over 80% of the requests are being executed by one drive pair. As mentioned, the workload has a typical VMS locality of reference. If the striping had been 100% effective, the striped curve would have yielded four times the requests per second of the unstriped. In fact, a gain of about two to one is shown indicating that the "hot spots" were not distributed completely. This in fact is typical of the gain that can be expected from striping.

Figure 11 shows the mean response time vs the I/O rate in requests per second for each RAID for large requests, four tracks per request. Again the workload was 25% writes and the chunk size for the RAID-0, RAID-1, RAID-5, and RAID-6 configurations is one track so that each request involved all drives.

With this workload and stripe size, RAID-0 and RAID-3 are good performers. They distribute the transfer and have no write overhead penalty. RAID-1 without striping is poor. It takes four revolutions for each transfer. RAID-1 with striping is still best. It offers the same advantages as RAID-0 and RAID-3 for single transfers and can do two read transfers simultaneously. RAID-5 and RAID-6 offer moderate performance. Again, they are penalized by the write algorithms.

Figure 11: Example Performance - Large Requests



CHAPTER 11

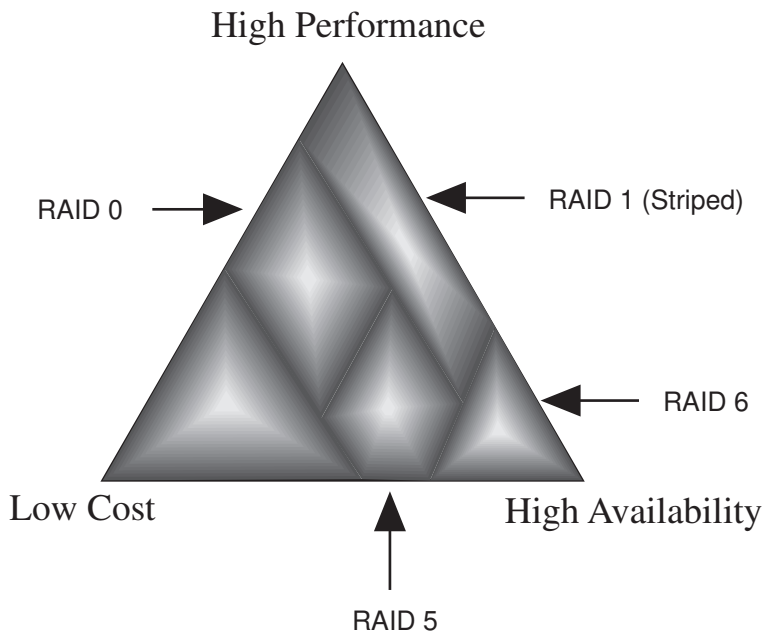
RAID VS USER REQUIREMENTS

Now let us return to the beginning. Figure 12 is the triangle shown in Figure 1. Please recognize that the scale of this figure is in the eye of the beholder. There are gray areas between the sections outlined within the triangle. It is intended to show that as you proceed across the bottom from left to right for example, a point is reached as you approach the middle where the user is willing to spend additional dollars to get additional availability. Likewise, going up the left side a point is reached as you approach the middle where the user is willing to spend extra dollars to get additional performance. Exactly where this point is or what rate the user is willing pay varies from user to user.

However, one can conclude as you move across the bottom that RAID-5 adds to the availability with the minimum additional cost. As one moves still further right, still more availability can be gained by implementing RAID-6, but at still more cost.

Moving up the left side of the triangle, availability is not important. One can gain performance with RAID-0 for very little cost. As one moves up higher still, adding greater importance to performance and less to cost, RAID-1 becomes the implementation of choice.

Figure 12: User Requirements - Small Requests



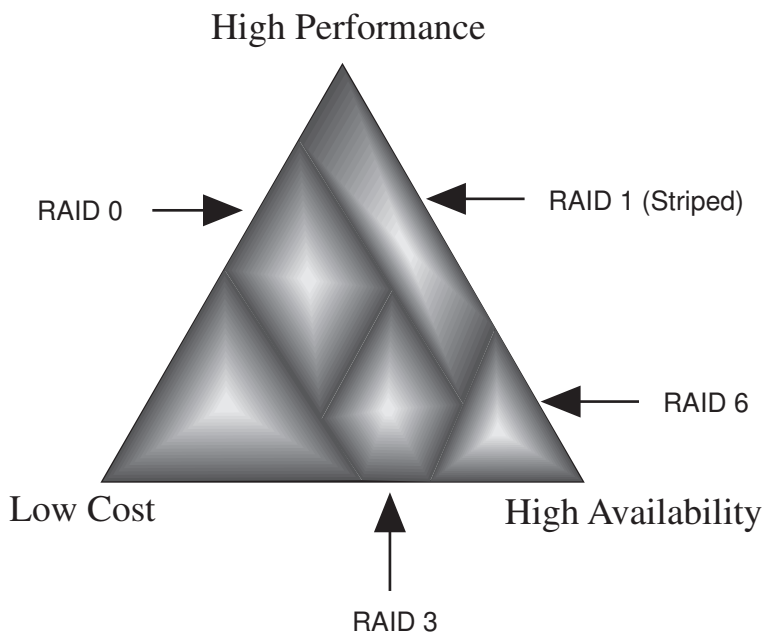
In the middle of the triangle is a RAID-0 to RAID-5 boundary. This division indicates the tradeoff between performance and availability.

Figure 13 is a similar figure for users with large request workloads. As you move across the bottom, RAID-3 adds to the availability with the minimum additional cost. As one moves still further right, still more availability can be gained by implementing RAID-6, but at still more cost.

Moving up the left side of the triangle, availability is not important. One can gain performance with RAID-0 for very little cost. As one moves up higher still, adding greater importance to performance and less to cost, RAID-1 becomes the implementation of choice.

In the middle of the triangle is a RAID-0 to RAID-3 boundary. This division indicates the tradeoff between performance and availability.

Figure 13: User Requirements - Large Requests



CHAPTER 12

CONCLUSIONS

It is hoped that the reader now has some basic understanding of what RAID is and recognizes that RAID is in fact a tradeoff between cost, performance, and availability. No single RAID level can satisfy the needs of all users. Each has attributes that make it a good solution for a particular set of needs. The reader should now be able to determine which configuration may best suit his particular needs.