Capítulo 14: Arrays redundantes de discos económicos (RAIDs)

En este capítulo introduciremos los RAIDs, una técnica que usa múltiples discos para construir un sistema de discos más rápido, grande y más fiable

Externamente un RAID se ve como una bestia compleja, consiste en muchos discos, mucha memoria y varios procesadores para gestionar el sistema. Un hardware del RAID es muy parecido a un ordenador y está especializado en la gestión del grupo de discos

Los RAIDs ofrecen un gran número de ventajas comparado con el uso individual de discos. Una ventaja es el rendimiento. Usar múltiples discos en paralelo puede acelerar enormemente los tiempos de I/O. Los RAID pueden mejorar la fiabilidad, difundir datos a través de múltiples discos hace que los datos sean vulnerables a pérdidas, pero haciéndolo de manera redundante los RAIDs pueden tolerar pérdidas de un disco y seguir funcionando como si nada hubiera pasado

Estas ventajas se las ofrece a los sistemas de manera transparente, éstos solo ven un disco duro grande en el sistema. Esto nos permite simplificar la sustitución de un disco duro por un sistema RAID sin tener que cambiar nada en el software

- Interfaz y elementos internos RAID

Cuando un archivo del sistema realiza una solicitud I/O al RAID, éste internamente tiene que calcular a qué disco o discos tiene que acceder para completar la solicitud y devolver una I/O física. Ésta dependerá del nivel del RAID

El sistema RAID normalmente está construido en un hardware separado del sistema que lo usará. En niveles altos, un RAID está tan especializado como un ordenador, con un procesador, memoria, discos… pero en vez de ejecutar aplicaciones, ejecuta software específico para operar el RAID

- Modelos de fallo

Los RAIDs están diseñados para detectar y recuperarse tras fallos en los discos, sabiendo qué fallos son críticos

El primer modelo de fallo es simple y es llamado fallo de parada. Un disco puede estar en dos estados, funcionando o fallido. Con un disco de trabajo todos los bloques pueden ser leídos o escritos. Cundo un disco está en el estado de fallo supondremos que todo se ha perdido para siempre

Un aspecto crítico de este modelo es que asume que hay detección de fallos. Cundo un disco falla asumimos que automáticamente se detecta por lo que no tendremos que preocuparnos por fallos silenciosos como corrupción en discos. No tendremos tampoco que preocuparnos porque un solo bloque se vuelva inaccesible dentro de un disco funcionando

- Cómo evaluar un RAID

La evaluación se llevará a cabo según tres factores. El primero es la capacidad. Teniendo N discos con B bloques cada uno, sin redundancia tendremos una capacidad de $N \cdot B$ bloques. Sin embargo, si tenemos un sistema que tiene dos copias de cada bloque obtenemos una capacidad de $(N \cdot B)/2$

El segundo factor es la fiabilidad. El número de discos perdidos que el sistema puede tolerar. Asumiremos que sólo un disco entero puede fallar, aunque más adelante veremos cómo manejar otros sistemas más complejos

El tercer factor es el rendimiento. Es difícil de calcular ya que depende de la carga de trabajo presente en cada disco del array. Antes de evaluar el rendimiento tendremos que tener presente sobre qué división estamos trabajando

Tendremos en cuenta tres diseños importantes de RAID: RAID 0 (división), RAID 1 (duplicación) y RAID 4/5 (redundancia basada en la paridad)

- RAID 0: División

Éste RAID no es realmente un RAID ya que no tenemos redundancia. Sin embargo, tiene un grandísimo rendimiento y capacidad

La manera más sencilla de dividir bloques en los discos del sistema es mediante un array tal que:

Disk 0	Disk 1	Disk 2	Disk 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Este enfoque está diseñado para obtener el mayor paralelismo del array cuando se realizan solicitudes para los trozos contiguos. Sin embargo, no siempre se da el caso de que solo se coloca un bloque por disco antes de pasar al siguiente, puede darse el caso de que se coloquen 2 o más bloques por disco antes de pasar al siguiente disco

Tamaños de trozos

El tamaño afecta al funcionamiento del array ya que uno pequeño significará que muchos archivos estarán divididos en varios discos, incrementando el paralelismo necesario para leer un solo archivo

Un tamaño grande significará que se reduce el paralelismo, por lo tanto, se basa en múltiples solicitudes concurrentes para lograr un alto rendimiento

Por lo tanto, determinar el mejor tamaño es difícil de hacer, ya que requiere un gran conocimiento sobre la carga de trabajo presentada al sistema de disco

Análisis del RAID 0

La capacidad es perfecta, teniendo N discos y B bloques tendremos una cantidad total de N·B bloques. La fiabilidad y el rendimiento también serán perfectos, pero de una mala manera: cualquier disco que falle dará lugar a la pérdida de datos

- Nivel RAID 1: Duplicación

Este sistema lo que hace es hacer una copia de cada bloque en el sistema, cada copia colocada en discos diferentes para conseguir tolerancia a fallos

En un ejemplo de sistema duplicado, asumimos que, para cada bloque lógico, el RAID tiene dos copias físicas de él

Disk 0	Disk 1	Disk 2	Disk 3
0	0	1	1
2	2	3	3
4	4	5	5
6	6	7	7

Esto es bastante común y a veces es llamado RAID-10 o RAID 1+0 porque usa la duplicación y el sistema de guardado del RAID 0

La lectura en estos casos puede ser realizada tanto en el original como en la copia, pero al escribir el RAID está obligado a actualizar ambos bloques para mantener la fiabilidad

Análisis RAID 1

Con N discos y B bloques, la capacidad del RAID 1 será de (N·B)/2 debido a los bloques duplicados. Sin embargo, este sistema es tolerante a fallos gracias a la duplicación. El número de discos fallidos que el sistema puede soportar, en los mejores casos, es de N/2, suponiendo que ningún disco contiguo se pierda

- RAID 4: ahorro de espacio con paridad

Ahora veremos un método diferente para añadir redundancia a un array de discos con la paridad. Esto nos permite usar menos capacidad para los datos duplicados, por lo que ganamos espacio útil para guardar datos, aunque tiene un pequeño coste

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3

Por cada fila de datos, añadimos un bloque de paridad que guarda información redundante sobre esos bloques. Para calcular la paridad usamos una función XOR. Es importante saber que el número de 1s en cualquier fila, incluyendo el bit de paridad, debe de ser un número par. El RAID debe mantener esto para que la paridad sea correcta

Podemos ver que el bit de paridad nos permite saber si hay algún error y recuperar los datos perdidos

- RAID 5: paridad rotativa

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
5	6	7	P1	4
10	11	P2	8	9
15	P3	12	13	14
P4	16	17	18	19

RAID 5 funciona prácticamente igual que RAID 4 pero con una diferencia, el bloque de paridad rota entre los discos