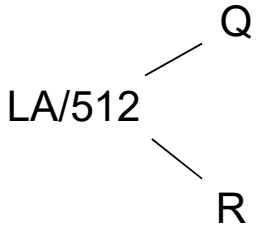
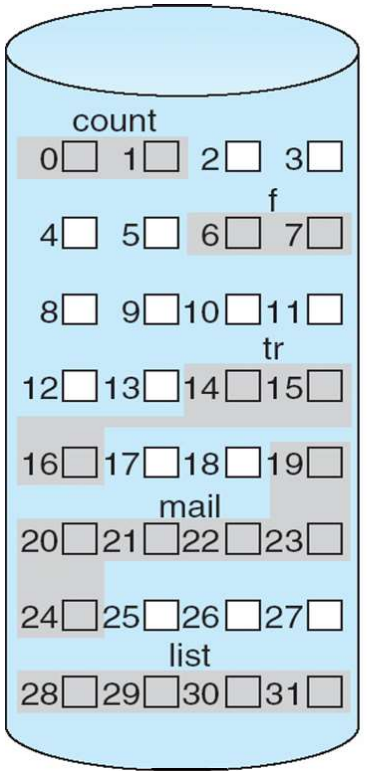
Allocation Methods - **Contiguous Allocation**

- Mapping from logical to physical



Block to be accessed = Q + starting address
Displacement into block = R

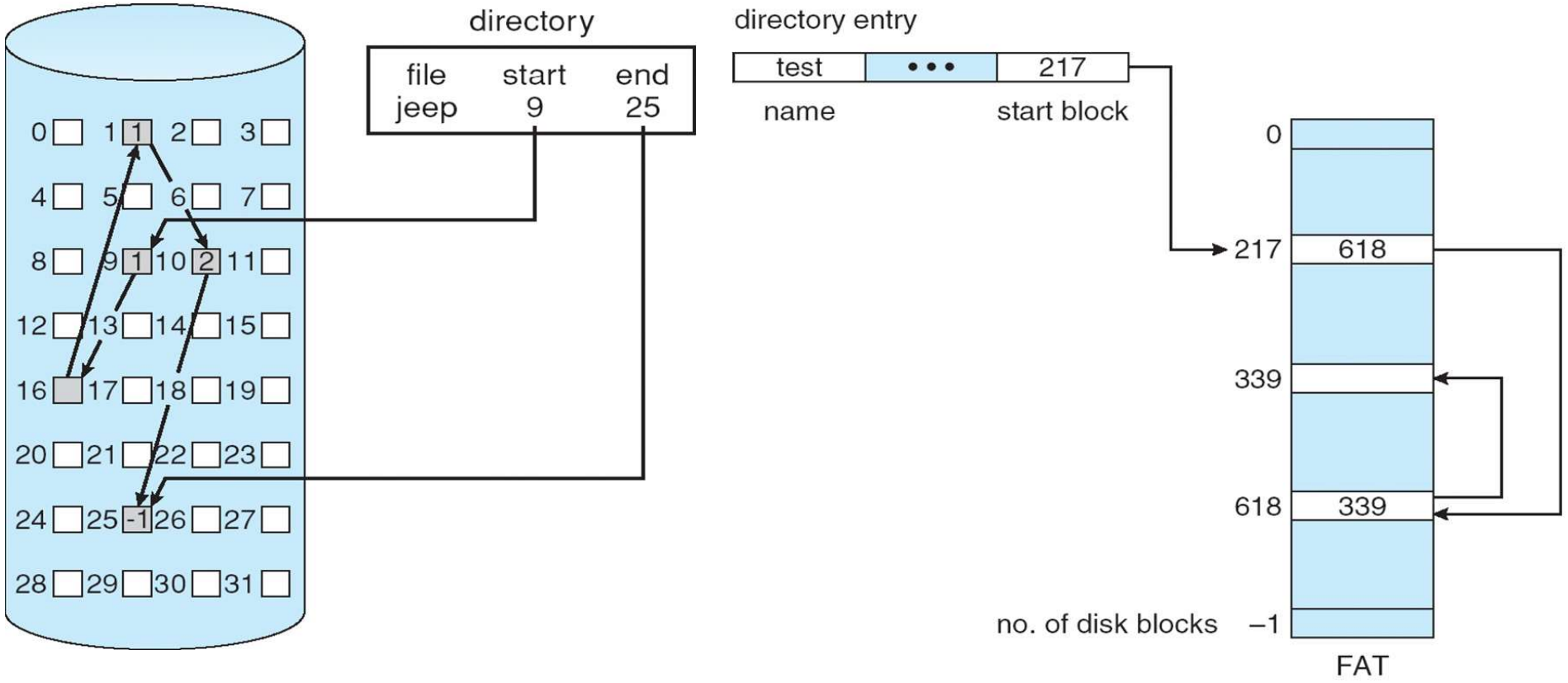


directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

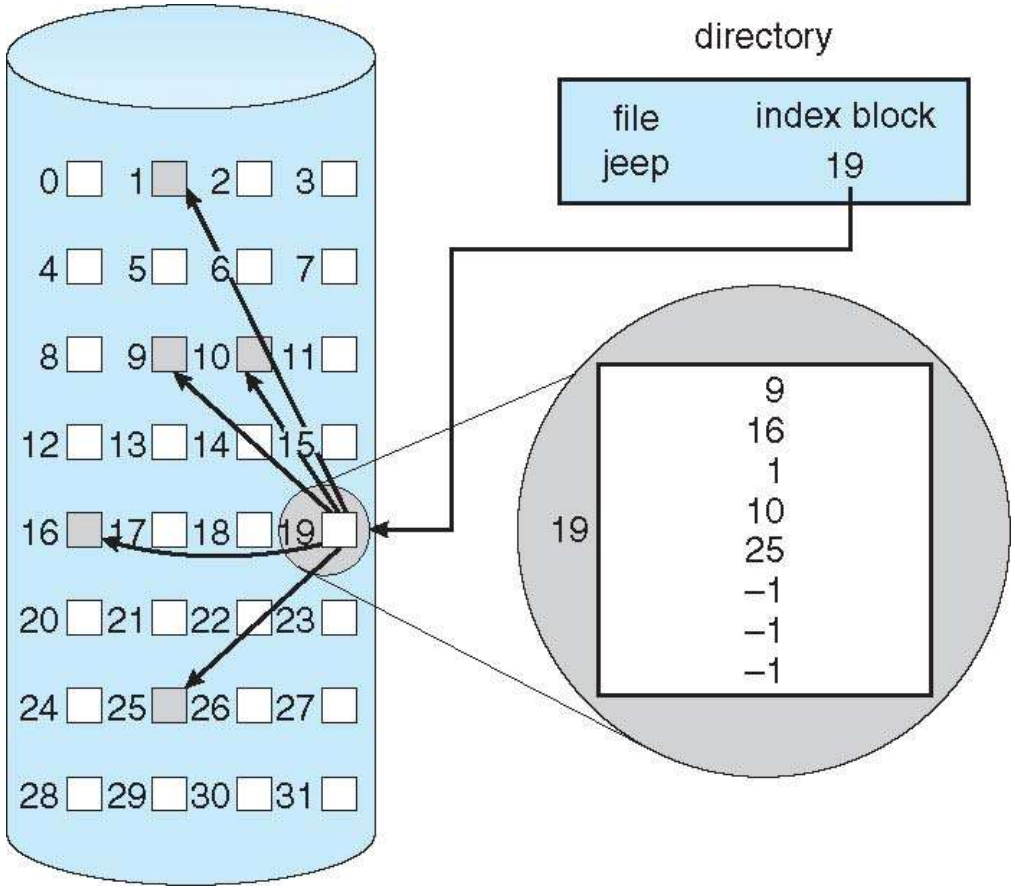
Best performance in most cases
Simple – only starting location (block #) and length (number of blocks) are required
Problems include finding space for file, knowing file size,

Allocation Methods - **Linked Allocation**



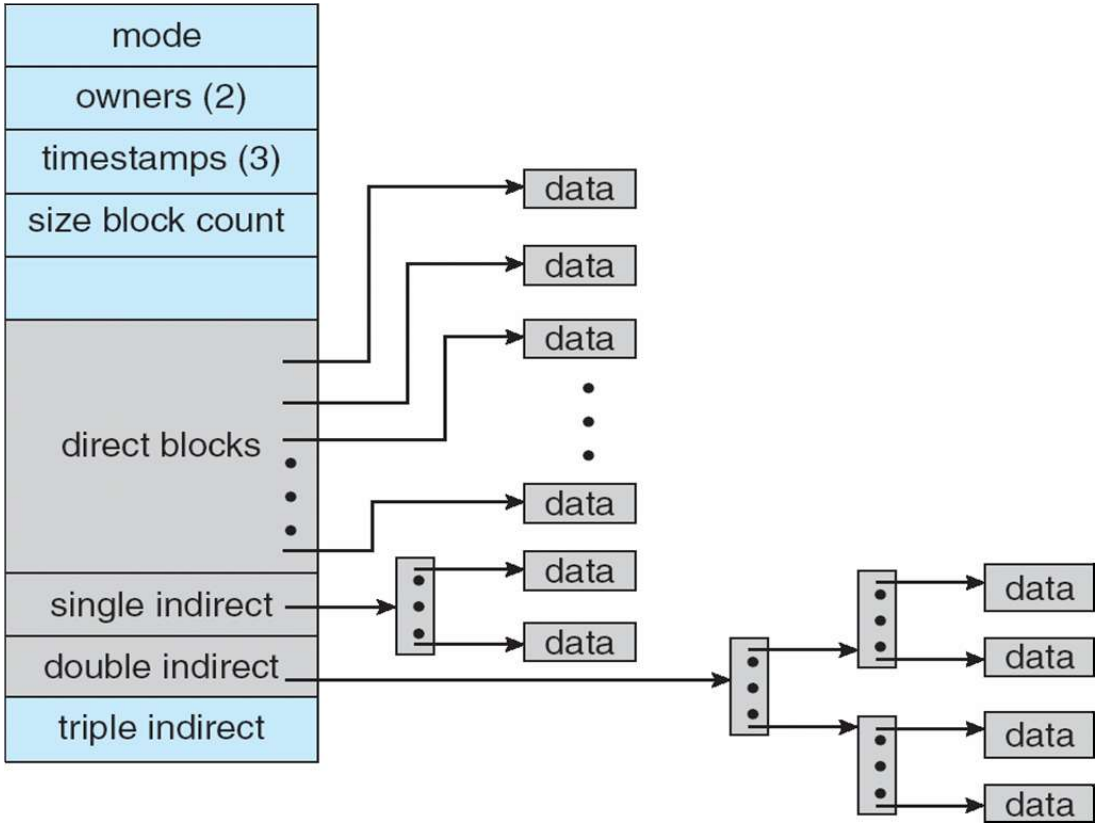
FAT (File Allocation Table) variation

Allocation Methods - **Indexed Allocation**



Combined Scheme: UNIX UFS

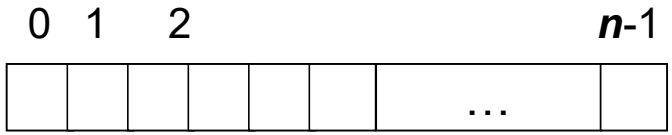
4K bytes per block, 32-bit addresses



More index blocks than can be addressed with 32-bit file pointer

Free-Space Management

- File system maintains **free-space list** to track available blocks/clusters
  - (Using term "block" for simplicity)
- **Bit vector** or **bit map** (***n*** blocks)



$$bit[i] = \begin{cases} 1 \Rightarrow block[i] \text{ free} \\ 0 \Rightarrow block[i] \text{ occupied} \end{cases}$$

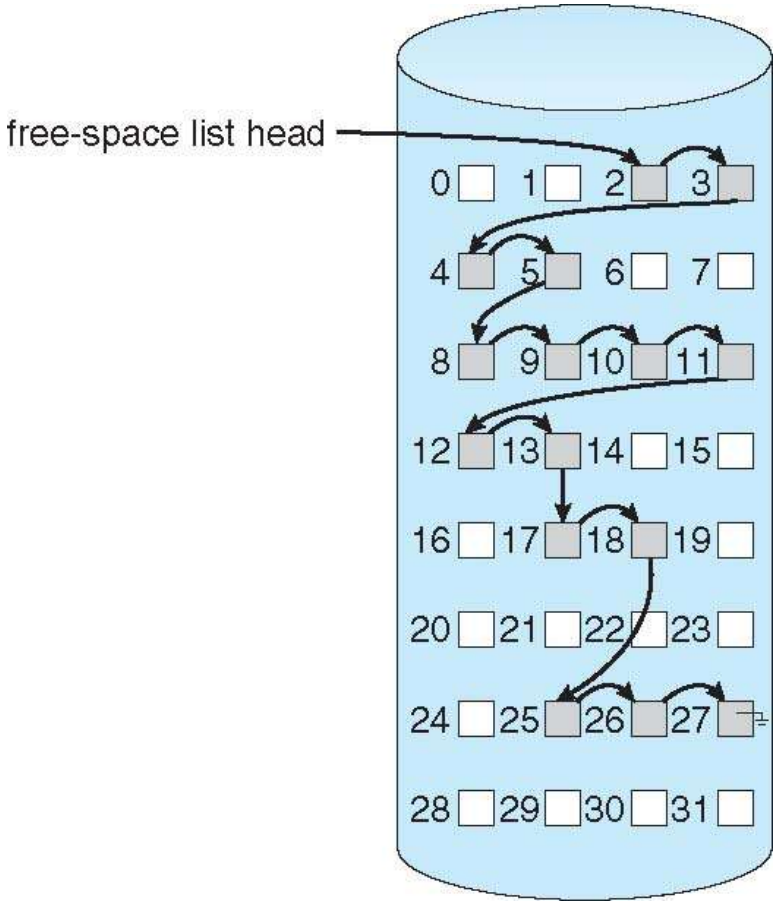
Block number calculation

(number of bits per word) *
(number of 0-value words) +
offset of first 1 bit

CPUs have instructions to return offset within word of first "1" bit

Linked Free Space List on Disk

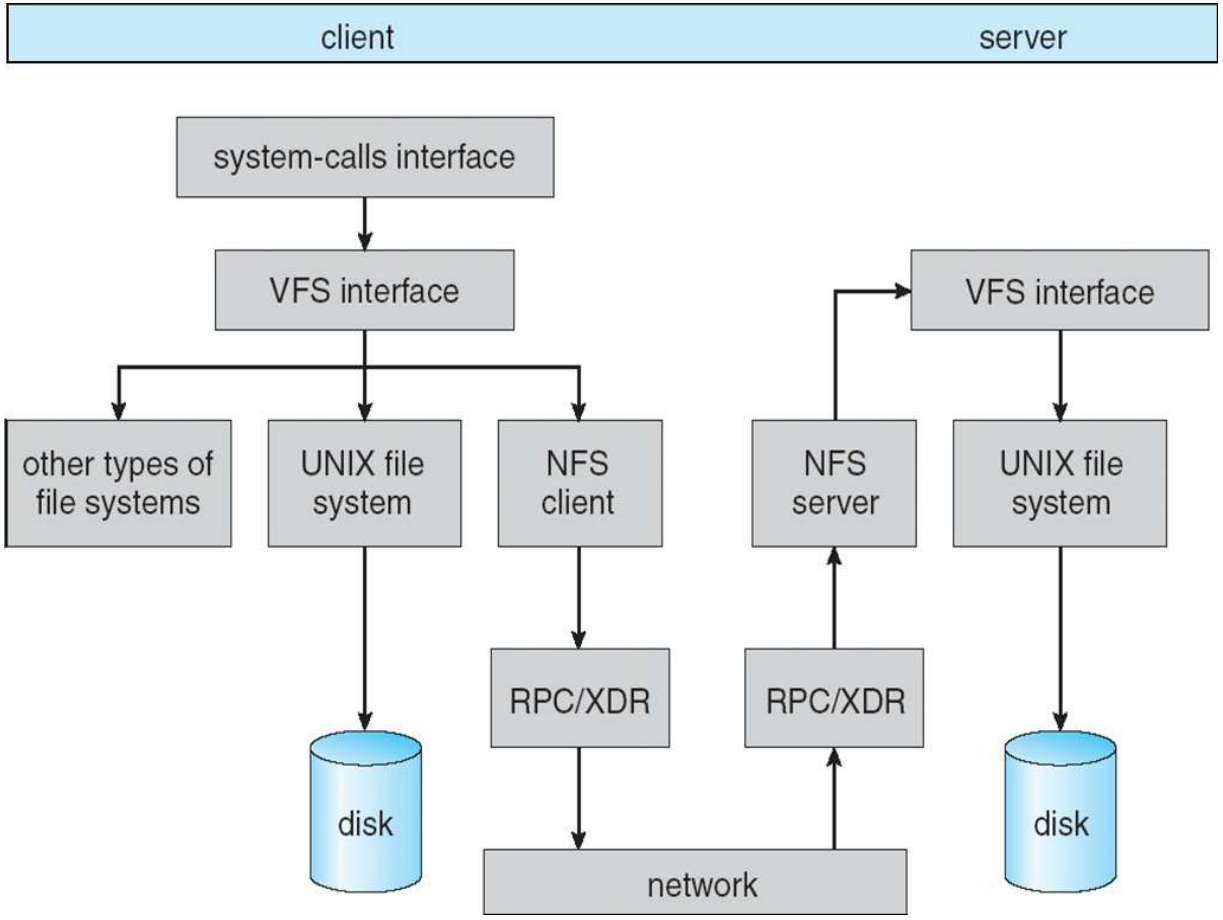
- Linked list (free list)
  - Cannot get contiguous s pace easily
  - No waste of space
  - No need to traverse the entire list (if # free blocks recorded)



Sun Network File System (**NFS**)

- An implementation and a specification of a software system for accessing remote files across LANs (or WANs)
- The implementation is part of the Solaris and SunOS operating systems running on Sun workstations using an unreliable datagram protocol (UDP/IP protocol and Ethernet

Schematic View of NFS Architecture



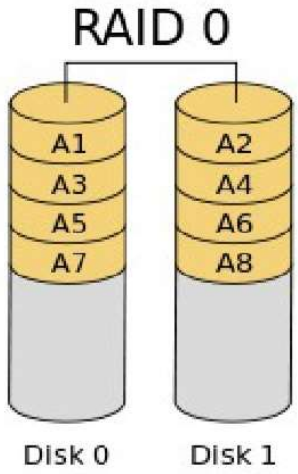
RAID Structure

- **RAID – redundant array of inexpensive disks**
  - multiple disk drives provides reliability via **redundancy**
- Increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors
- If mirrored disks fail independently, consider disk with 1300,000 mean time to failure and 10 hour mean time to repair
  - Mean time to data loss is $100,000^2 / (2 * 10) = 500 * 10^6$ hours, or 57,000 years!
- Frequently combined with **NVRAM** to improve write performance
- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively

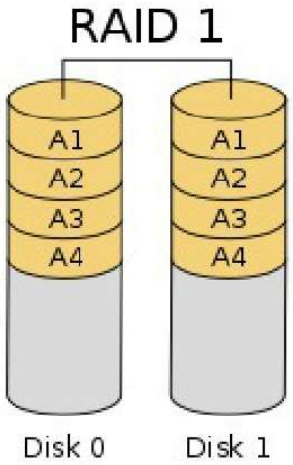
RAID (Cont.)

- Disk **striping** uses a group of disks as one storage unit
- RAID is arranged into six different levels
- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data
  - **Mirroring** or **shadowing** (**RAID 1**) keeps duplicate of each disk
  - Striped mirrors (**RAID 1+0**) or mirrored stripes (**RAID 0+1**) provides high performance and high reliability
  - **Block interleaved parity** (**RAID 4, 5, 6**) uses much less redundancy
- RAID within a storage array can still fail if the array fails, so automatic **replication** of the data between arrays is common
- Frequently, a small number of **hot-spare** disks are left unallocated, automatically replacing a failed disk and having data rebuilt onto them

RAID Levels

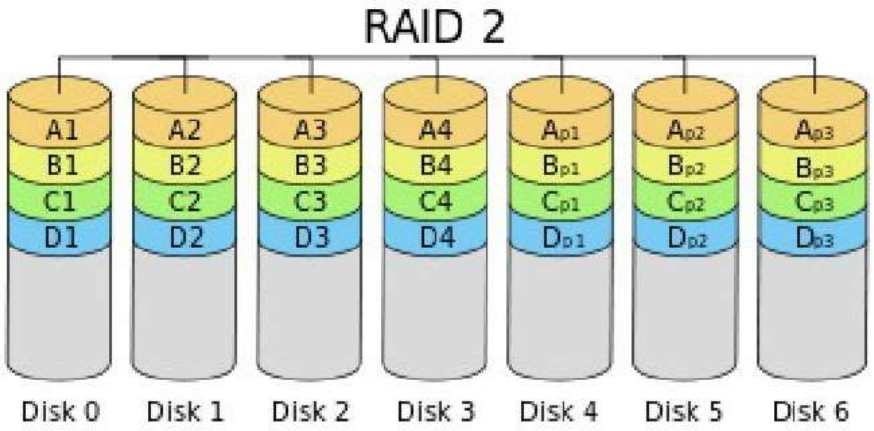


Striping 이라고도 부르는 방식.
RAID 0를 구성하기 위해서는 최소 2개의 디스크가 필요(min(N) == 2).
RAID를 구성하는 모든 디스크에 데이터를 분할하여 저장.
전체 디스크를 모두 동시에 사용하기 때문에 성능은 단일 디스크의 성능의 N배.
하나의 디스크라도 문제가 발생 할 경우 전체 RAID가 깨지는 문제가 발생.

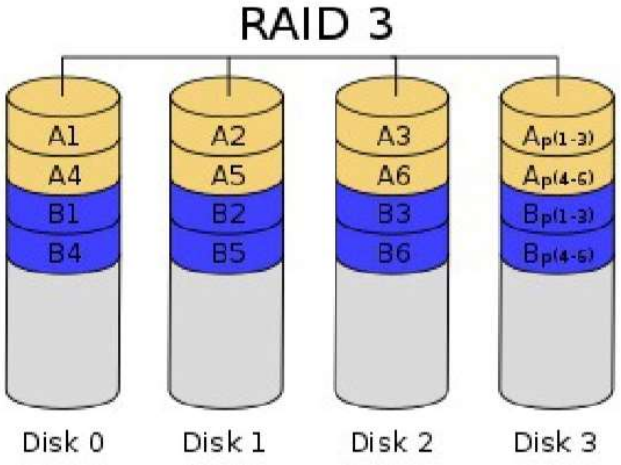


Mirroring 이라고도 부르는 방식.
RAID 1을 구성하기 위해서는 최소 2개의 디스크가 필요(min(N) == 2).
RAID 컨트롤러에 따라서 2개의 디스크로만 구성 가능할 수도,
  그 이상의 개수를 사용 하여 구성 할 수도 있음.
RAID 1은 모든 디스크에 데이터를 복제하여 기록.
  동일한 데이터를 N개로 복제하여 각 디스크에 저장.
  실제 사용 가능한 용량은 단일 디스크의 용량과 동일.
N-1개의 디스크가 고장 나도 데이터 사용이 가능한 안정성을 제공.
  비용 문제로 인해 거의 사용하지 않음.

RAID Levels

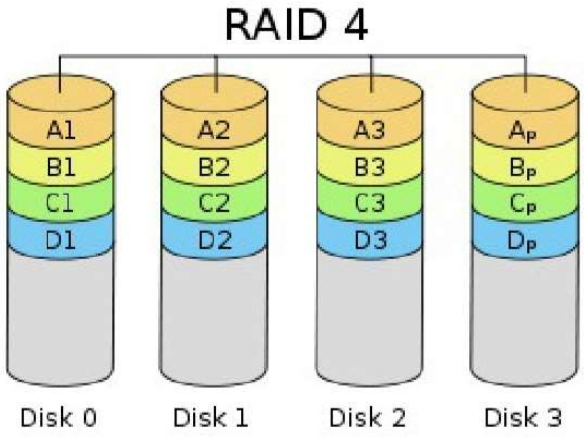


현재는 사용하지 않는 RAID Level.
bit 단위로 striping을 하고,
error correction을 위해 **Hamming code**를 사용.

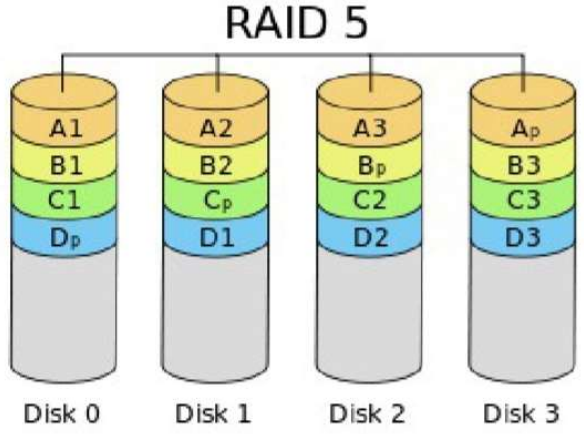


현재는 사용하지 않는 RAID Level.
Byte 단위로 striping을 하고,
Byte 단위로 striping하기 때문에 너무 작게 쪼개져 사용하지 않음.
error correction을 위해 **패리티 디스크**를 1개 사용.
용량 및 성능이 단일 디스크 대비 (N-1) 배 증가.
최소 3개의 디스크로 구성 가능 - 1개의 디스크 에러 시 복구 가능.

RAID Levels

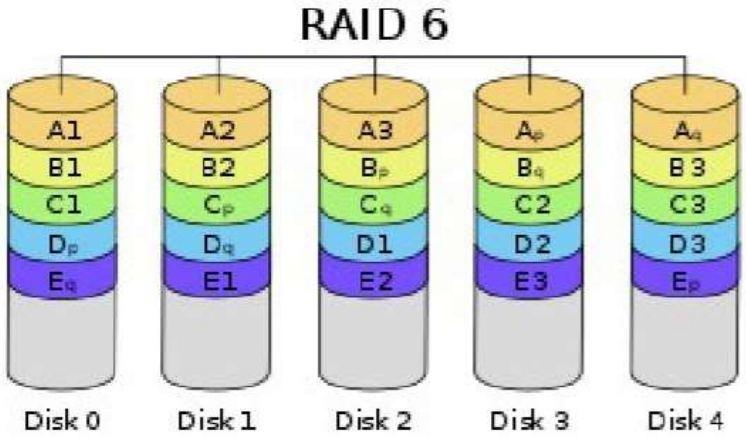


현재는 거의 사용하지 않는 RAID Level.
Block 단위로 striping을 하고,
error correction을 위해 **패리티 디스크**를 1개 사용.
용량 및 성능이 단일 디스크 대비 (N-1) 배 증가.
최소 3개의 디스크로 구성 가능 - 1개의 디스크 에러 시 복구 가능.
Block 단위로 striping 하는 것은 RAID 5, RAID 6와 동일하지만,
  패리티 코드를 **동일한 디스크에 저장**하기 때문에,
  패리티 디스크의 사용량이 높아 해당 디스크의 수명이 줄어듬.
RAID 4의 단점을 개선시킨 것이 RAID 5.



제일 사용 빈도가 높은 RAID Level.
Block 단위로 striping을 하고,
error correction을 위해 **패리티를 1개의 디스크에 저장**.
  패리티 저장 하는 디스크를 고정하지 않고,
  매 번 **다른 디스크에 저장**.
용량 및 성능이 단일 디스크 대비 (N-1) 배 증가.
최소 3개의 디스크로 구성 가능 - 1개의 디스크 에러 시 복구 가능.
RAID 0에서 성능, 용량을 조금 줄이는 대신 안정성을 높인 RAID Level.

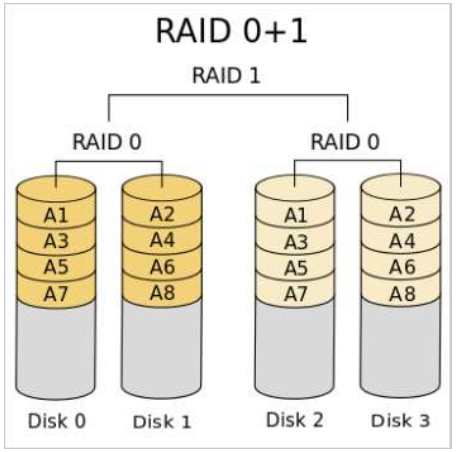
RAID Levels



RAID 5서 성능, 용량을 더 줄이고, 안정성을 더 높인 RAID Level
안정성을 높여야 하는 서버 환경에서 주로 사용.
Block 단위로 striping을 하고,
error correction을 위해 **패리티를 2개의 디스크에 저장**.
패리티 저장 하는 디스크를 고정하지 않고,
매 번 다른 디스크에 저장.
용량 및 성능이 단일 디스크 대비 (N-2) 배 증가.
최소 4개의 디스크로 구성 가능 - 2개 디스크 에러 시 복구 가능.

Nested(중첩) RAID

Nested RAID는 Standard RAID를 여러 개 중첩하여 사용. 즉, 복수의 Standard RAID를 RAID로 묶는 것.



(1) 2개의 RAID 0를 RAID 1으로 묶음
(RAID 0+1 혹은 RAID 01)

(2) 2개의 RAID 1을 RAID 0로 묶을 수 있음.
(RAID 1+0 혹은 RAID 10)