RAID-МАСИВИ

Питання 4.2

Структура RAID

- Маючи велику кількість носіїв інформації в системі, можна покращити швидкість зчитування або запису даних, якщо носії працюватимуть паралельно.
  - Так можна підвищити надійність зберігання, дублюючи дані на різних носіях.
  - Для цього широко використовуються методи організації дискового простору, які колективно називають ***надмірними масивами незалежних дисків (redundant arrays of independent disks, RAID).***
- У минулому RAID-масиви компонувались з маленьких, дешевих дисків та розглядались як cost-effective альтернатива великим та дорогим дискам.
  - Нині RAID-масиви використовуються для кращої надійності та вищої швидкості передачі даних, ніж з економічних причин.
  - Тому ***I*** в абревіатурі ***RAID,*** що раніше означало “inexpensive”, тепер значить “independent.”

Покращення надійності за допомогою надмірності

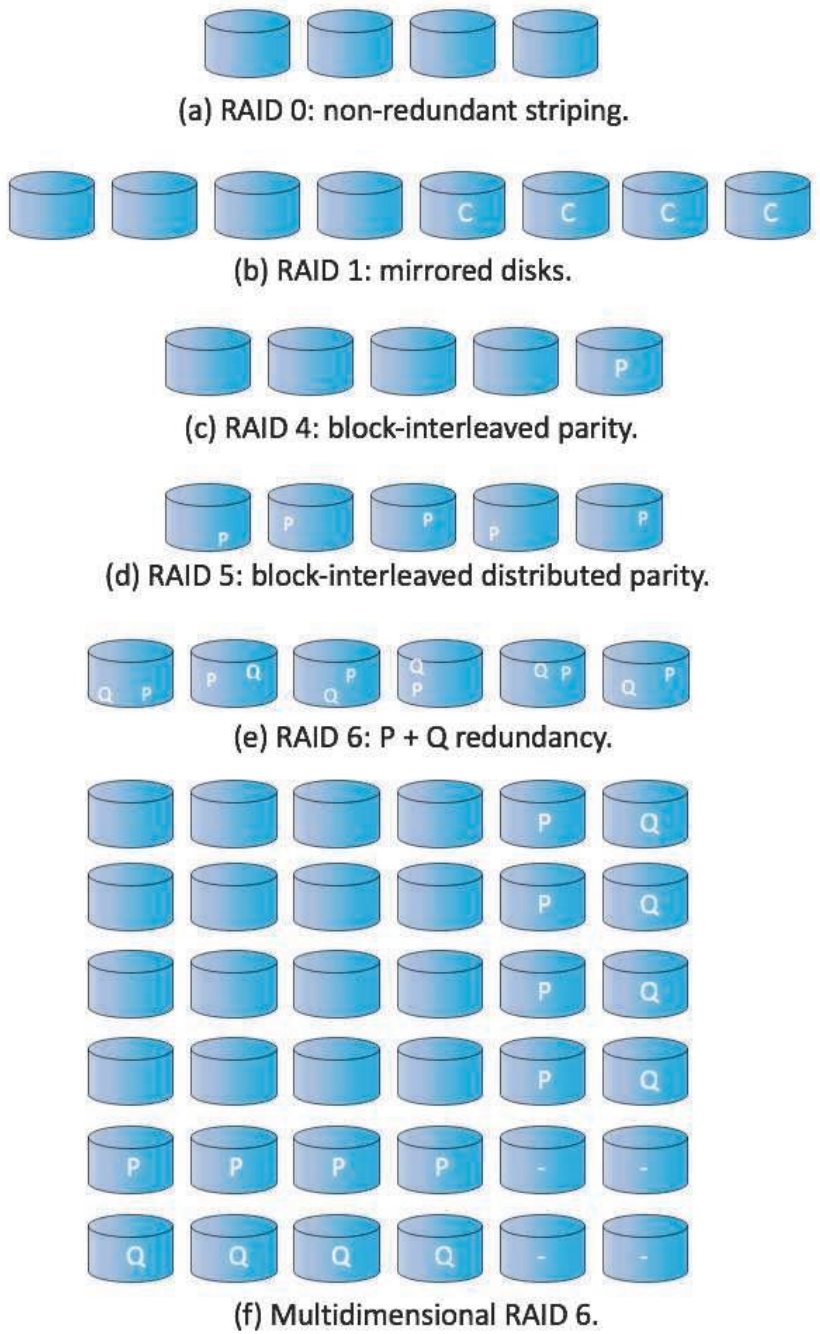
- Спочатку розглянемо RAID з жорстких дисків.
  - Ймовірність відмови диску з набору *N* дисків набагато вища, ніж ймовірність відмови окремого диску.
  - Нехай середній наробіток між відмовами ***(mean time between failures, MTBF)*** одного диску складає 100000 годин.
  - Тоді MTBF деякого диску в масиві зі 100 дисків буде 100000/100 = 1000 годин, тобто 41.66 дня!
  - Якщо зберігаємо тільки одну копію даних, кожна відмова диску буде спричиняти втрату значної частини даних—така швидкість втрати даних неприйнятна.
- Вирішення проблеми – ***надмірне зберігання (redundancy)***: зберігаємо додаткову інформацію, яка у звичайному стані непотрібна, проте може бути використана в випадку відмови диску для відновлення втраченої інформації.
  - RAID може застосовуватись і до NVM-носіїв, хоч NVM-накопичувачі менш схильні до відмови.
- Найпростіший, проте найдорожчий спосіб впровадження надмірності – дублювання кожного носія.
  - Такий підхід називають ***дзеркалюванням (mirroring)***. Логічний диск складається з 2 фізичних носіїв, а кожний запис виконується на обидва пристрої.
  - Утворюються ***віддзеркалені томи***. Дані будуть втрачені лише за відмови обох томів.
- MTBF віддзеркаленого тому залежить від 2 чинників.
  - 1) MTBF окремих носіїв. 2) Середній час ремонту – заміну збійного носія та відновлення даних на новому.
  - Нехай відмови 2 носіїв незалежні, MTBF одного накопичувача – 100000 годин, а середній час ремонту - 10 годин.
  - Середній час втрати даних віддзеркаленої системи накопичувачів: $100000^2/(2 * 10) = 500 * 10^6$ годин або ***57000 років***!

Проте неможливо припускати незалежність відмови пристроїв

- Втрата електропостачання чи природні катастрофи можуть призвести до пошкодження обох накопичувачів одночасно.
  - Дефекти при виробництві в партіях пристроїв можуть спричинити корельовані відмови.
  - Старіння накопичувачів теж підвищує ймовірність їх відмови, зокрема й відмови дзеркала, поки ремонтується основний носій.
- Втрата електропостачання – найбільша причина хвилювання, оскільки природні катастрофи відбуваються не настільки часто.
  - Навіть при дзеркалюванні, якщо запис відбувається одночасно в один і той же блок на обох носіях, втрата електропостачання до повного запису обох блоків може призвести до неузгодженого їх стану.
  - Одне з вирішень – спочатку записати одну копію, а потім – іншу.
  - Інше – додати твердотільний енергонезалежний кеш у RAID-масив.
  - Кеш відкладеного запису (write-back cache) захищається від втрати даних при відключенні електроенергії, тому запис може вважатись завершеним за умови наявності для кешу деякого захисту від помилок, а також методів їх корекції, зокрема ECC або дзеркалювання.

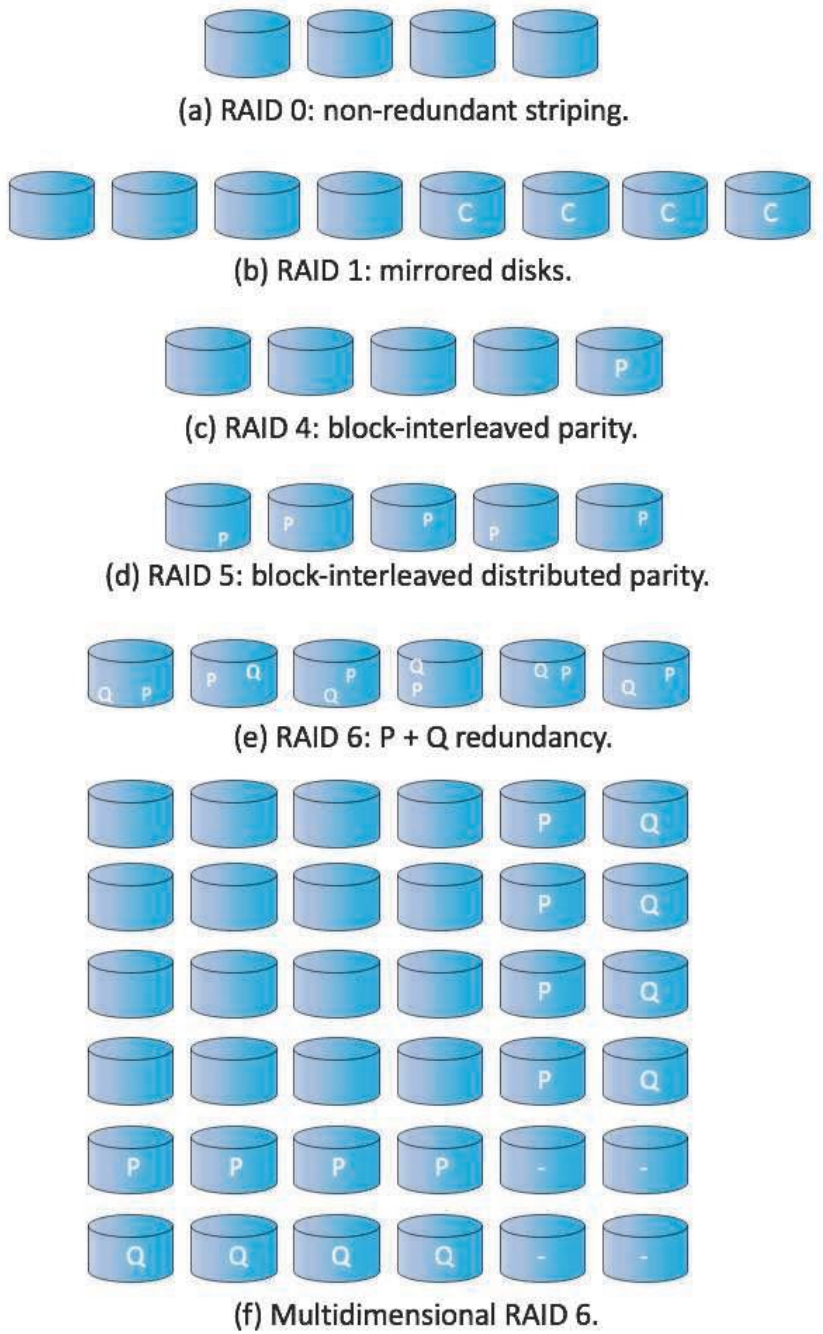
Покращення продуктивності за допомогою паралелізму

- З дзеркалюванням швидкість обробки запитів на зчитування може подвоїтись.
  - Швидкість передачі зчитаних даних залишається тією ж.
- Маючи багато накопичувачів, можемо покращити transfer rate, чергуючи дані з пристроїв.
  - У найпростішій формі ***чергування даних (data striping)*** складається з розподілу бітів кожного байту між накопичувачами – ***чергування на рівні бітів (bit-level striping)***.
  - Наприклад, маючи масив з 8 накопичувачів, записуємо біт *i* кожного байту на пристрій *i.*
  - Такий масив може розглядатись як один великий носій з секторами, що у 8 разів більші за звичайний розмір, а також мають у 8 разів вищу швидкість доступу.
  - Кожний накопичувач бере участь у кожному доступі (читанні чи запису).
- Чергування на рівні бітів можна узагальнити для кількості носіїв – дільників або кратних 8.
  - Наприклад, для масиву з 4 пристроїв, біти *i* та 4+*i* кожного байту надходять на носій *i.*
  - У подальшому чергуванні на рівні бітів потреби немає.
- ***Чергування на рівні блоків (block-level striping)*** передбачає чергування блоків файлу між *n* накопичувачами: блок *i* йде на пристрій (*i* mod *n*)+1.
  - Інші рівні чергування, на зразок байтів сектору чи секторів блоку, також можливі.
  - Чергування на рівні блоків – єдине широкодоступне чергування.



Рівні RAID

- Дзеркалювання дає високу надійність, проте дороге; чергування забезпечує високу швидкість передачі даних, проте не покращує надійність.
  - Були запропоновані численні схеми для забезпечення надмірності найменшою ціною, використовуючи чергування дисків у комбінації з бітами парності – ***рівні RAID***.
  - Вони мають різні компроміси ціна-продуктивність.
- **RAID level 0**. Масиви накопичувачів з чергуванням даних на рівні блоків, проте без надмірності (дзеркалювання чи бітів парності).
- **RAID level 1**. RAID level 1 – дзеркалювання носіїв.
- **RAID level 4**. RAID level 4, також відомий як *memory-style error-correcting code (ECC) organization*. Має виділений диск парності. ECC також використовується в RAID 5 та 6.



Ідея ECC може напряму використовуватись у масивах зберігання даних через чергування блоків між носіями

- Наприклад, перший блок даних для запису може записуватись на диск 1, другий блок – на диск 2 і т.д. до *N*-того блоку; результат обчислення ECC для цих блоків запишеться на диск *N* + 1 (пункт (c) на рисунку).
  - Відмова одного диску призведе до переобчислення ECC, запобігання передачі даних запитуючому процесу та викидання помилки.
- RAID 4, насправді, може коригувати помилки, хоч присутній лише один блок ECC.
  - Контролери накопичувачів можуть визначати коректність зчитування, тому один блок парності можна застосувати для виявлення та корекції помилок.
  - Якщо один з секторів пошкоджений, точно відомо, що це за сектор.
  - Відкидаємо дані з цього сектору та використовуємо парні дані, щоб заново обчислити пошкоджені дані.
  - Для кожного біту в блоці можна визначити, повинен він бути 1 чи 0, обчислюючи парність відповідних бітів з секторів на інших носіях.
  - Якщо парність решти бітів дорівнює збереженій парності, відсутній біт буде 0, інакше – 1.

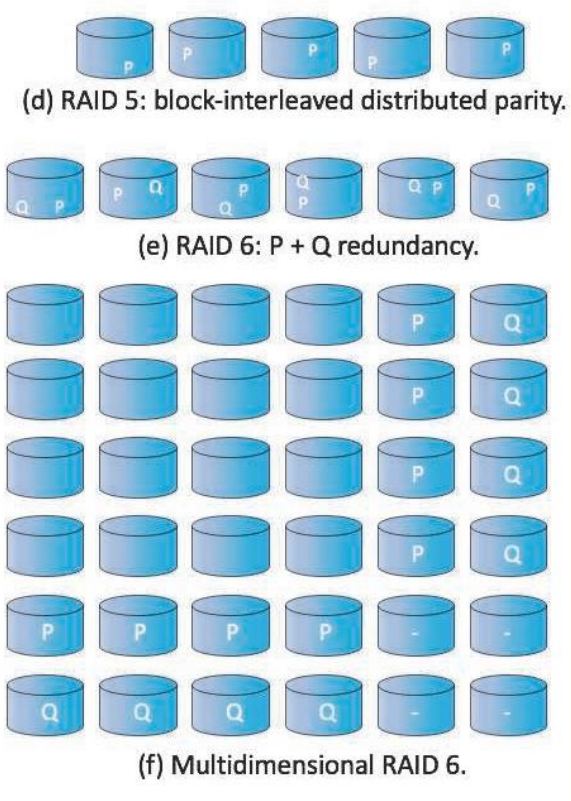
RAID level 4

- Зчитування блоку виконує доступ тільки до одного носія, дозволяючи іншим запитам оброблятись іншими накопичувачами.
  - Швидкість передачі для зчитування великих обсягів висока, оскільки всі диски можна зчитувати паралельно.
  - Записування великих обсягів даних також має високу швидкість, оскільки дані та блоки парності можна записувати паралельно.
- Невеличкі незалежні записування не можуть виконуватись паралельно.
  - Записування ОС даних, менших за розмір блоку, вимагає, щоб цей блок було зчитано, оновлено новими даними з їх подальшим записом.
  - Також потрібно оновити блок парності. Цей процес називають ***циклом read-modify-write***.
  - Тому одна операція запису вимагає 4 доступи до диску: 2 на зчитування старих блоків та 2 на запис нових блоків.
- RAID level 4 має 2 переваги над level 1, забезпечуючи рівномірну проєкцію даних.
  - 1) Накладні витрати збереження зменшуються, оскільки тільки 1 диск парності потрібний для кількох звичайних накопичувачів.
  - 2) оскільки зчитування та записування ряду блоків поширюється на багато накопичувачів у зв'язку з чергуванням даних *N* способами, швидкість передачі для зчитування або запису набору блоків у *N* разів вища, ніж для level 1.

Проблема продуктивності RAID 4

- Як і для всіх RAID-рівнів з підтримкою парності, дорожнеча обчислення та запису XOR-парності.
  - Ці накладні витрати можуть призвести до сповільнення запису в порівнянні з RAID-масивами без парності.
- Сучасні ЦП загального призначення дуже швидкі в порівнянні з дисковим вводом-виводом, проте втрату продуктивності можна мінімізувати.
  - Багато RAID-масивів або host bus-adapters включають апаратний контролер з виділеним обчислювачем парності.
  - Масив також має NVRAM-кеш, щоб зберігати блоки в процесі обчислення парності та буферизувати дані для запису від контролера до накопичувачів.
  - Така буферизація дозволяє уникати більшість циклів read-modify-write, збираючи дані для записування у стрічку (full stripe) та конкурентно записуючи відповідні частини на всі накопичувачі.
  - Комбінація апаратного прискорення та буферизації може зробити RAID-масиви з парністю майже такими ж швидкими, як RAID-масиви без парності, часто навіть перевершуючи некешовані RAID-масиви без парності.

RAID level 5 та 6



- Для наочності парність RAID показана на виділених дисках, проте насправді RAID-блоки розкидані по таблиці.

RAID level 5 (block-interleaved розподілена парність) відрізняється від level 4 тим, що поширює дані та парність серед усіх *N*+1 накопичувачів.

- Для кожного набору з *N* блоків один з дисків зберігає парність, а решта – дані. Наприклад, для 5 дисків парність з *n*-того блоку зберігається в накопичувачі (*n* mod 5) + 1. *n*-тий блок решти дисків містить дані.
- Блок парності не може зберігати парність для блоків на одному і тому ж носії, оскільки його відмова спричинить втрату даних і парності, тому відновлення інформації буде неможливим.
- Поширюючи парність на всі накопичувачі, RAID 5 уникає потенційного надмірного використання одного диску парності, характерного для RAID 4. RAID 5 – найбільш поширений рівень RAID з парністю.

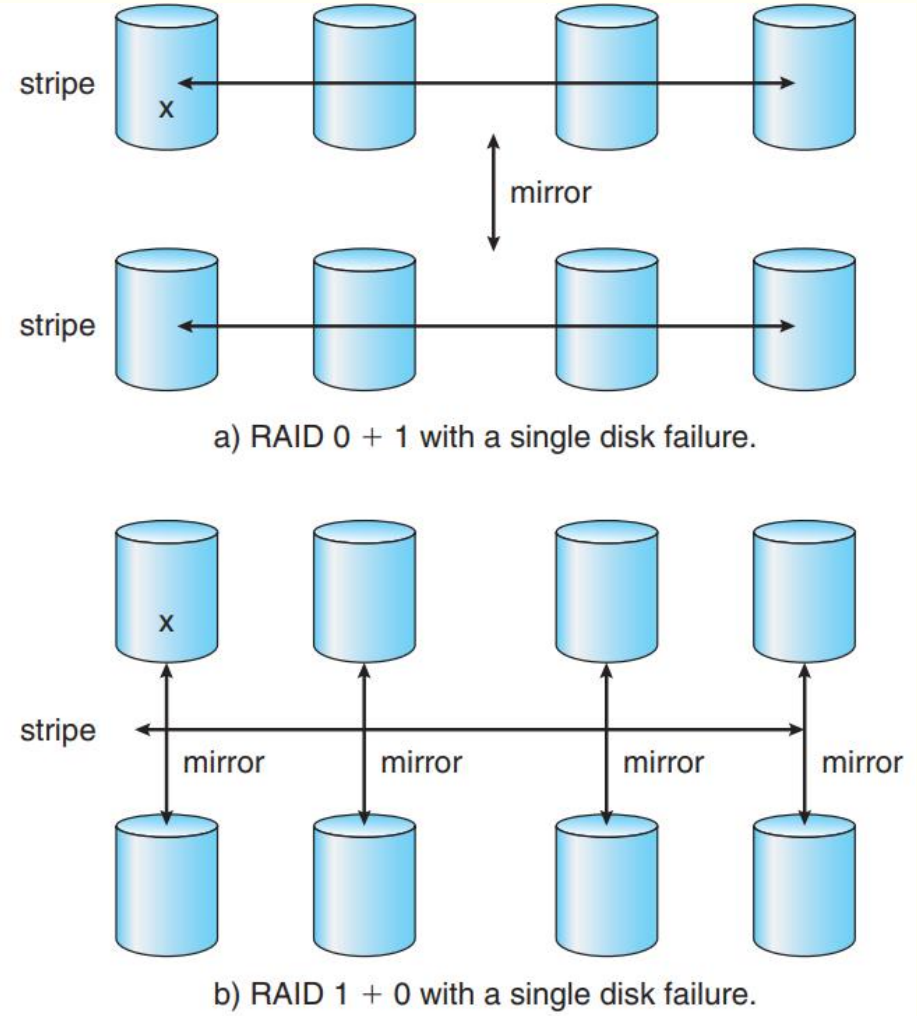
RAID level 6 («схема надмірності P+Q») багато в чому схожий на RAID level 5, проте з зберігає додаткову надмірну інформацію для захисту від відмови багатьох накопичувачів.

- XOR-парність не може застосовуватись on both parity blocks, оскільки вони будуть ідентичними та не нададуть інформації для відновлення даних. Замість парності використовуються ECC, зокрема на базі полів Галуа, для обчислення Q.
- 2 блоки надмірних даних зберігаються для кожних 4 блоків даних, і система може пережити відмову 2 дисків.

Багатовимірний RAID level 6. Деякі складні масиви зберігання даних посилюють RAID 6.

- Нехай масив містить сотні накопичувачів. Зібравши їх у стрічку RAID level 6, отримаємо багато дисків з даними та тільки 2 логічних диски парності.
- Багатовимірний RAID level 6 логічно впорядковує диски в таблицю та реалізує RAID level 6 як по горизонталі, так і по вертикалі. Систему можна відновити практично від будь-якої відмови.

RAID levels 0 + 1 and 1 + 0



- **RAID level 0 + 1** – комбінація RAID 0 та 1.
  - RAID 0 забезпечує продуктивність, а RAID 1 – надійність.
  - Загалом даний рівень забезпечує кращу швидкодію, ніж RAID 5.
  - На жаль, як RAID 1 він подвоює кількість дисків для зберігання, тому також є відносно дорогим.
  - У RAID 0 + 1 набір накопичувачів збираються у стрічку, яка потім віддзеркалюється.
- **RAID level 1 + 0** – накопичувачі віддзеркалюються попарно, а потім віддзеркалені пари збираються в стрічку.
  - Схема має деякі теоретичні переваги в порівнянні з RAID 0 + 1.
  - Наприклад, якщо один носій відмовляє в RAID 0 + 1, вся стрічка стає недоступною.
  - Така відмова у RAID 1 + 0 робить недоступним тільки 1 диск, проте решта дисків і дзеркало залишаються доступними.

До базових RAID-схем пропонувались численні варіації

- Реалізація RAID – ще одна область для варіювання.
  - ***ПЗ для управління дисковим простором може реалізувати RAID на рівні ядра або системного ПЗ***. Тоді апаратне сховище може забезпечити мінімальні features та все ще бути частиною повного RAID-рішення.
  - ***RAID може реалізуватись у хост-адаптері шини (HBA)***. Тільки напряму підключені до HBA накопичувачі можуть бути частиною заданого RAID-масиву. Рішення дешеве, проте не дуже гнучке.
  - ***RAID може реалізуватись апаратно в масиві зберігання даних***. Масив зберігання даних може створити RAID-набори різних рівнів та навіть розбивати (slice) їх на менші томи, представлені операційній системі.
    - ОС потрібно тільки реалізувати файлову систему на кожному з томів.
    - Масиви мають багато доступних підключень або можуть бути частиною SAN, дозволяючи багатьом хостам отримувати переваги від можливостей масиву.
  - ***RAID може реалізовуватись у шарі інтерконнекту SAN за допомогою drive virtualization devices***.
    - Тоді пристрій стає між хостами та сховищем.
    - Він приймає команди від серверів та управляє доступом до сховища.
    - Він може забезпечити, наприклад, дзеркалювання, записуючи кожний блок на 2 окремих накопичувача.

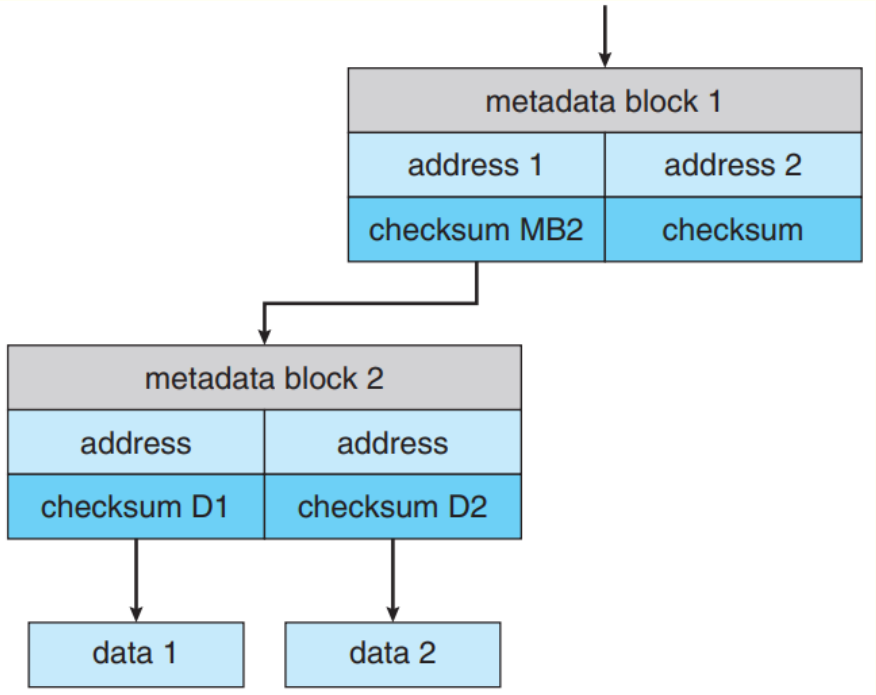
Інші можливості, що можна реалізувати на кожному з цих рівнів

- ***Знімок файлової системи (snapshot)* –** представлення ФС до того, як відбулось останнє оновлення.
- ***Реплікація*** включає автоматичне дублювання записаних даних між окремими вузлами (sites) for redundancy та disaster recovery.
  - Реплікація може бути синхронною або асинхронною.
  - При синхронній реплікації кожний блок повинен записуватись локально та віддалено до того, як запис вважатиметься завершеним, а при асинхронній реплікації записування групуються та періодично записуються.
  - Асинхронна реплікація може призвести до втрати даних, якщо основний вузол (site) відмовляє, проте вона швидша та не має обмежень по відстані. Використання реплікації також зростає в датацентрах чи навіть на хості.
  - Як альтернатива до RAID-захисту, реплікація захищає від втрати даних, а також підвищує продуктивність зчитування (дозволяючи читати з кожної репліки). Очевидно, що реплікація використовує більше простору, ніж більшість рівнів RAID.
- Реалізація цих можливостей різниться залежно від прошарку, на якому реалізується RAID.
  - Наприклад, якщо RAID реалізується програмно, кожному хосту може знадобитись створення та керування власної replication.
  - Якщо реплікація реалізована у масиві зберігання даних або в інтерконнекті SAN, тоді незалежно від ОС хоста або його можливостей дані хоста можуть реплікуватись.
- Інший аспект більшості реалізацій RAID – ***диски гарячої заміни (hot spare drives)***.
  - Гаряча заміна не використовується для даних, проте is configured to be used as a replacement in case of drive failure.
  - For instance, a hot spare can be used to rebuild a mirrored pair should one of the drives in the pair fail.
  - In this way, the RAID level can be reestablished automatically, without waiting for the failed drive to be replaced.
  - Allocating more than one hot spare allows more than one failure to be repaired without human intervention.

Вибір рівня RAID

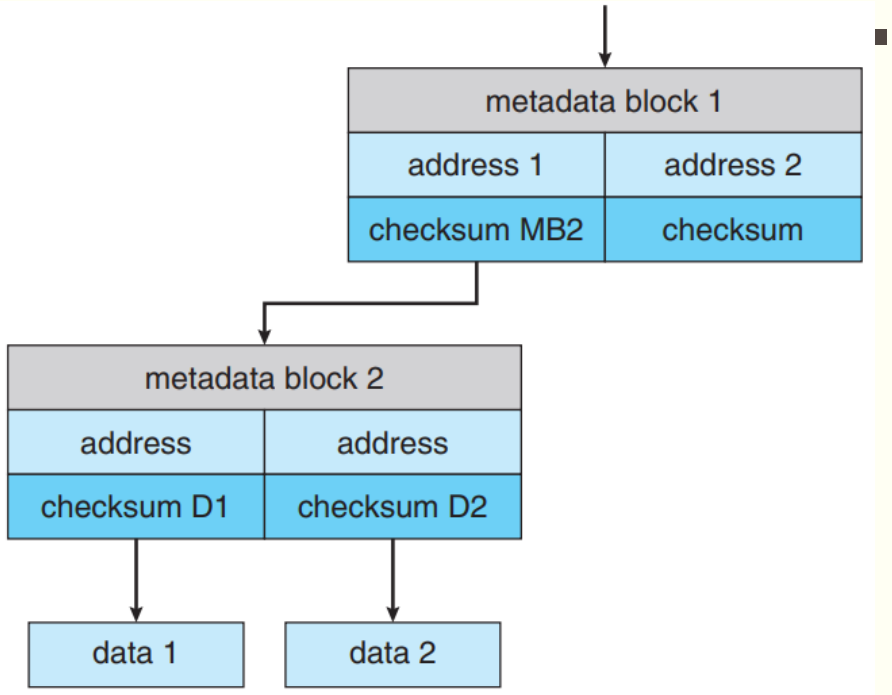
- ***Продуктивність відновлення даних (rebuild performance)***. Якщо накопичувач відмовляє, необхідний час to rebuild даних може бути значним.
  - Важливий фактор за потреби неперервного постачання даних, as it is in high-performance or interactive database systems.
  - Furthermore, rebuild performance influences the mean time between failures.
- Rebuild performance варіюється залежно від рівня RAID.
  - Rebuilding найпростіше для RAID level 1, since data can be copied from another drive.
  - For the other levels, we need to access all the other drives in the array to rebuild data in a failed drive. Rebuild times can be hours for RAID level 5 rebuilds of large drive sets.
- RAID level 0 is used in high-performance applications where data loss is not critical.
  - For example, in scientific computing where a data set is loaded and explored, RAID level 0 works well because any drive failures would just require a repair and reloading of the data from its source.
  - RAID level 1 is popular for applications that require high reliability with fast recovery.
  - RAID 0 + 1 and 1 + 0 are used where both performance and reliability are important—for example, for small databases.
  - Due to RAID 1's high space overhead, RAID 5 is often preferred for storing moderate volumes of data.
  - RAID 6 and multidimensional RAID 6 are the most common formats in storage arrays. They offer good performance and protection without large space overhead.
- RAID system designers and administrators of storage have to make several other decisions as well.
  - For example, how many drives should be in a given RAID set? How many bits should be protected by each parity bit?
  - If more drives are in an array, data-transfer rates are higher, but the system is more expensive.
  - If more bits are protected by a parity bit, the space overhead due to parity bits is lower, but the chance that a second drive will fail before the first failed drive is repaired is greater, and that will result in data loss.

Проблеми з RAID



- RAID не завжди забезпечує доступність даних для ОС та її користувачів.
  - Вказівник на файл або у file structure, наприклад, можуть бути неправильними.
  - Неповні записування (torn writes), якщо не відновлені коректно, можуть призвести до пошкодження даних.
  - Також деякі інші процеси можуть випадково виконати запис поверх структур файлової системи.
  - RAID захищає від помилок фізичного середовища, проте не від інших апаратних чи програмних помилок.
  - Відмова апаратного RAID-контролеру або баг у відповідному програмному коді може призвести до повної втрати даних.
- Файлова система **Solaris ZFS** застосовує інноваційний підхід для вирішення проблем за допомогою контрольних сум.
  - ZFS підтримує внутрішні контрольні суми всіх блоків, включаючи дані та метадані.
  - Ці контрольні суми не тримаються з блоком, для якого вони рахуються. Частіше вони зберігаються із вказівником на цей блок.

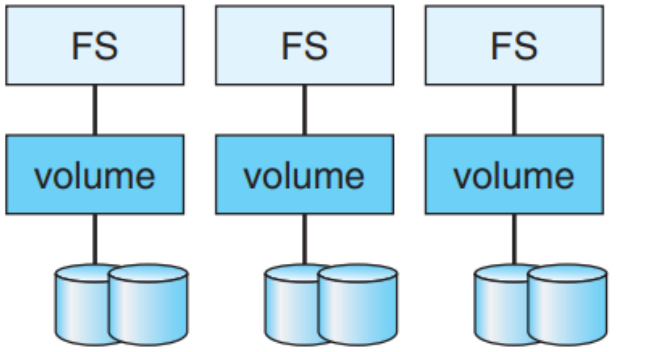
Проблеми з RAID



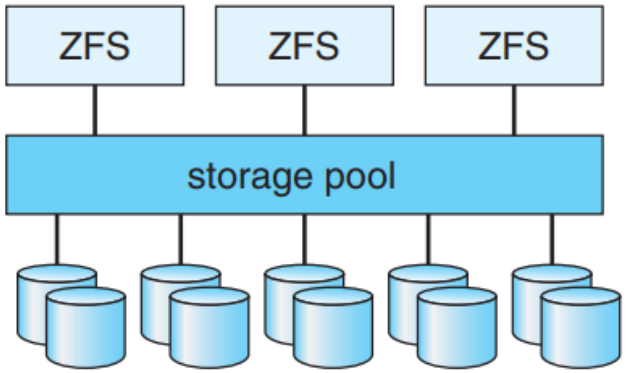
- Розглянемо **inode** – структуру даних для зберігання метаданих файлової системи – із вказівниками на дані ФС.
  - Всередині inode є контрольна сума кожного блоку даних.
  - Якщо є проблема з даними, контрольна сума буде некоректною, а файлова система дізнається про це.
  - Якщо дані віддзеркалені, і присутній відповідний блок з коректною контрольною сумою, ZFS автоматично оновить битий блок хорошими даними.
  - Аналогічно, directory-запис, який вказує на inode, має контрольну суму для inode. Будь-яка проблема в inode виявляється при доступі до directory.
  - Такі перевірки відбуваються по всіх структурах ZFS, надаючи набагато вищий рівень узгодженості, виявлення та виправлення помилок, ніж у наборах RAID-накопичувачів чи стандартних ФС.
  - Додаткові накладні витрати, створені обчисленням контрольних сум та додаткові цикли read-modify-write для блоків непомітні, оскільки загальна швидкодія ZFS дуже висока.

Інша проблема з більшістю реалізацій RAID – нестача гнучкості

- Нехай масив зберігання даних з 20 накопичувачів розбито на 4 набори по 5 носіїв.
  - Кожний набір є набором RAID level 5: 4 окремих томи мають власну файлову систему.
  - Що, якщо одна файлова система надто велика для підтримки на 5-дисковому RAID level 5 наборіset?
  - І якщо інша файлова система потребує дуже мало місця?
  - Якщо ці фактори відомі заздалегідь, накопичувачі та томи можна ретельно алокувати. Проте дуже часто використання пристрою та вимоги до нього змінюються з часом.
- Навіть якщо масиву зберігання даних дозволено створити весь набір з 20 дисків як один величезний RAID-масив, можуть виникнути інші проблеми.
  - Кілька томів різного обсягу можуть увійти в набір. Проте деякі менеджери дискового простору не дозволяють змінювати розмір тому.
  - У такому випадку залишимось з неузгодженими розмірами файлових систем.
  - Деякі менеджери дискового простору дозволяють змінювати розмір, проте деякі файлові системи не допускають свого розширення чи зменшення.
  - Томи можуть змінювати розмір, проте файлові системи потрібно буде заново створювати, щоб отримати переваги від цих змін.



(a) Traditional volumes and file systems.



(b) ZFS and pooled storage.

- ZFS комбінує керування файловою системою та управління дисковим простором у єдиний засіб (unit).
- Накопичувачі або їх розділи збираються за допомогою RAID sets у ***пули (pools)*** зберігання. Пул може тримати кілька файлових систем ZFS.
  - Вільний простір всього пулу доступний всім файловим системам всередині цього пулу.
- ZFS використовує модель пам'яті функцій malloc() та free() для виділення та очищення сховища для кожної файлової системи при використанні та очищенні блоків у ній.
  - У результаті немає штучних обмежень на використання сховища та потреби в переміщенні файлових систем між томами чи зміни розміру томів.
  - ZFS надає квоти для обмеження розміру ФС та зарезервовані області пам'яті для розростання ФС на визначений обсяг, проте ці змінні власник ФС може змінювати в будь-який момнт.
  - Інші системи, як Linux, мають менеджери дискового простору, які дозволяють логічно поєднувати багато дисків, щоб створити larger-than-disk томи для тримання великих файлових систем.