Основы построения файловых систем

Сегодня мы поговорим про RAID

RAID – Redundant Array of Independent (Inexpensive) Disks

Для чего нужен:

- Большая надёжность, чем у отдельных дисков,
- Большая вместимость, чем у отдельных дисков.

Статистика Backblaze по поломкам 4TB HDD в 2015 году

Модель	Число дисков	% поломавшихся
HGST Deskstar 5K4000	2600	0.86%
HGST Megascale 4000	7000	0.70%
Seagate Desktop HDD.15	20900	3.31%

https://www.backblaze.com/blog/hard-drive-reliability-q3-2015/

Эксперимент в CERN о надёжности хранения данных

- Приложение пишет 1Gb данных на диск следующим образом:
  - Записать 1Mb,
  - Подождать 1с,
  - Повторить.
- Запускаем такое приложение на каждом из дисков на кластере из 3000 машин с HW RAID.
- Через 3 недели читаем содержимое файлов.

Эксперимент в CERN о надёжности хранения данных

- Приложение пишет 1Gb данных на диск следующим образом:
  - Записать 1Mb,
  - Подождать 1с,
  - Повторить.
- Запускаем такое приложение на каждом из дисков на кластере из 3000 машин с HW RAID.
- Через 3 недели читаем содержимое файлов.
- Нашлось примерно 150 одномегабайтных блоков с изменившимся содержимым, причём чтение из них завершалось «успешно» с точки зрения как оборудования, так и файловой системы.

Эксперимент в CERN о надёжности хранения данных

Выводы:

- Данные нельзя хранить в единственном экземпляре,
- Необходимы чексуммы для проверки целостности,

- Необходима активная фоновая проверка данных.

Эксперимент в CERN о надёжности хранения данных

Выводы:

- Данные нельзя хранить в единственном экземпляре,
- Необходимы чексуммы для проверки целостности,
- Необходима активная фоновая проверка данных.
- Хранение реплик или использование Reed-Solomon,
- ZFS и btrfs хранят криптографические хеши всех записанных данных, ext4 хранит только CRC,
- Online scrubbing & repair в ZFS и btrfs или в HW RAID-контроллерах.

Оценки надёжности HDD и SSD

- MTBF – Mean Time Between Failures
- RBER – Raw Bit Error Rate
- UBER – Uncorrectable Bit Error Rate

Оценки надёжности HDD и SSD

- MTBF – Mean Time Between Failures
- RBER – Raw Bit Error Rate
- UBER – Uncorrectable Bit Error Rate

Для SSD S3710 Intel обещает

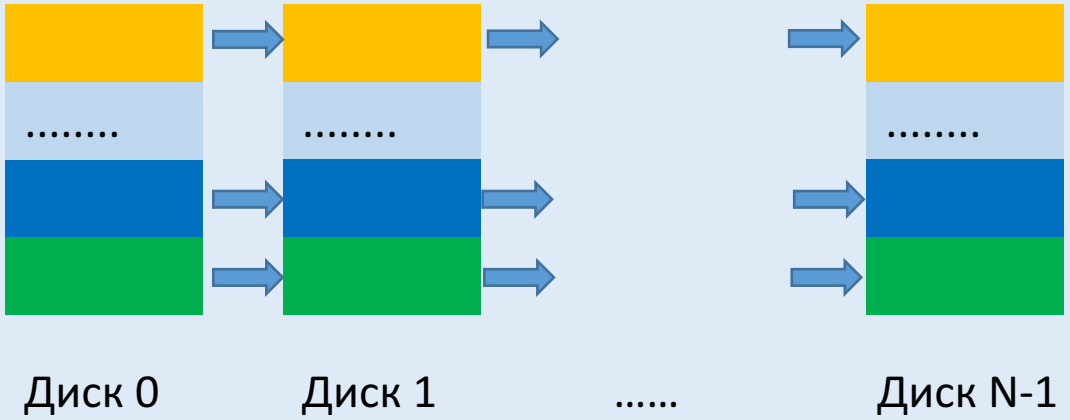
- Reliability
  - Uncorrectable Bit Error Rate (UBER):
    1 sector per 10^17 bits read
  - Mean Time Between Failures (MTBF): 2 million hours
  - End-to-End data protection

Означают ли эти числа, что в реальной жизни ошибки нам не встретятся?
Что будет в ДЦ объёмом 10PB, где стоит тысяча дисков?

Уровни RAID

RAID0 (stipe)

Данные разрезаются на последовательные куски длины N * B, каждый кусок разделяется на N частей, которые записываются на различные диски:



Уровни RAID

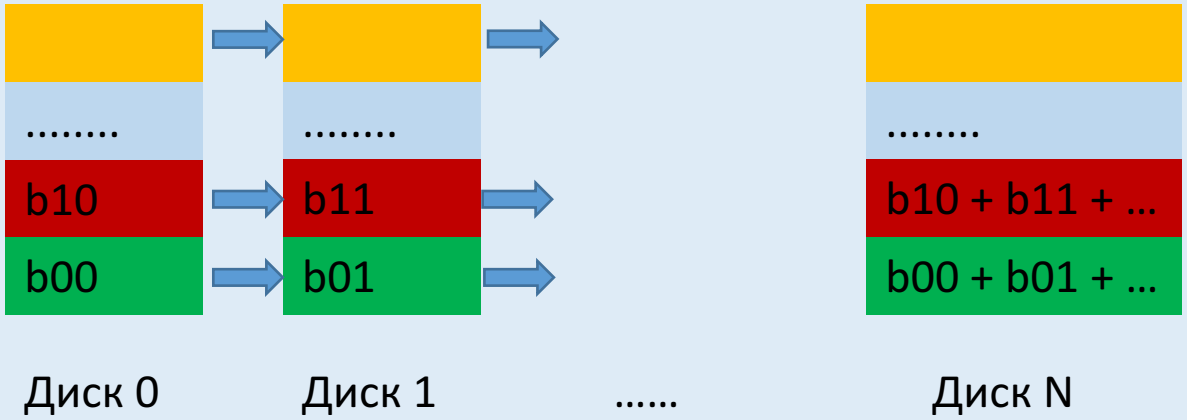
RAID1 (mirror)

Каждый диск в массиве содержит одни и те же данные:

Уровни RAID

RAID4

Массив состоит из N+1 дисков. На первых N дисках данные хранятся, как на RAID0.
На последнем диске каждый блок вычисляется как XOR соответствующих блоков на N дисках.



При потере любого диска массив остаётся работоспособным.

Уровни RAID

RAID4

Массив состоит из N+1 дисков. На первых N дисках данные хранятся, как на RAID0.
На последнем диске каждый блок вычисляется как XOR соответствующих блоков на N дисках.

При потере любого диска массив остаётся работоспособным.

Такой массив имеет концептуальный недостаток: диск с блоками чётности будет изнашиваться быстрее других дисков.

Уровни RAID

RAID5

Массив строится так же, как и RAID4, но блоки чётности в разных страйпах хранятся на разных дисках:

Уровни RAID

RAID0 (stripe)	• большая скорость линейных записи и чтения, • оптимизирует случайные чтения, • теряет все данные при поломке одного диска.
RAID1 (mirror)	• оптимизирует случайное чтение, • скорость записи – как у одиночного диска, • выживает при потере всех дисков, кроме одного, • слишком расточителен.
RAID5	• оптимизирует случайное чтение, • **что со скоростью записи?** • позволяет потерять любой диск без потери работоспособности.
RAID6	Как RAID5, но вычисляет два разных блока чётности, поэтому выдерживает потерю любых двух дисков.
RAID10	Пары дисков объединяются в RAID1, затем поверх этих RAID1 собирается RAID0.

Трудности с RAID

- (небольшая) При перезагрузке устройства могут поменять имена.
- (большая) Write holes.

Write holes

Запись на разные диски будет происходить в разное время.

Рассмотрим такой сценарий:

- начинается запись на RAID1,
- диск #0 обработал запрос на запись сектора,
- произошёл сбой питания,
- на диске #1 сектор остался без изменений.

Write holes

Аппаратный способ решения:

- BBU (Battery Backup Unit) в RAID-контроллерах.

Программные способы решения:

- write intent bitmap (linux md),
- checksumming + COW (ZFS),
- SSD journal: https://lwn.net/Articles/665299/ .

Write intent bitmap, помимо исправления write holes, позволяет уменьшить время проверки и перестроения массива после аварийного выключения.

Скорость записи на RAID5

Из-за необходимости переживать аварийные выключения мы не имеем права одновременно изменять несколько блоков в одном страйпе. Значит, скорость записи на RAID5 получается такая же, как на одиночный диск.

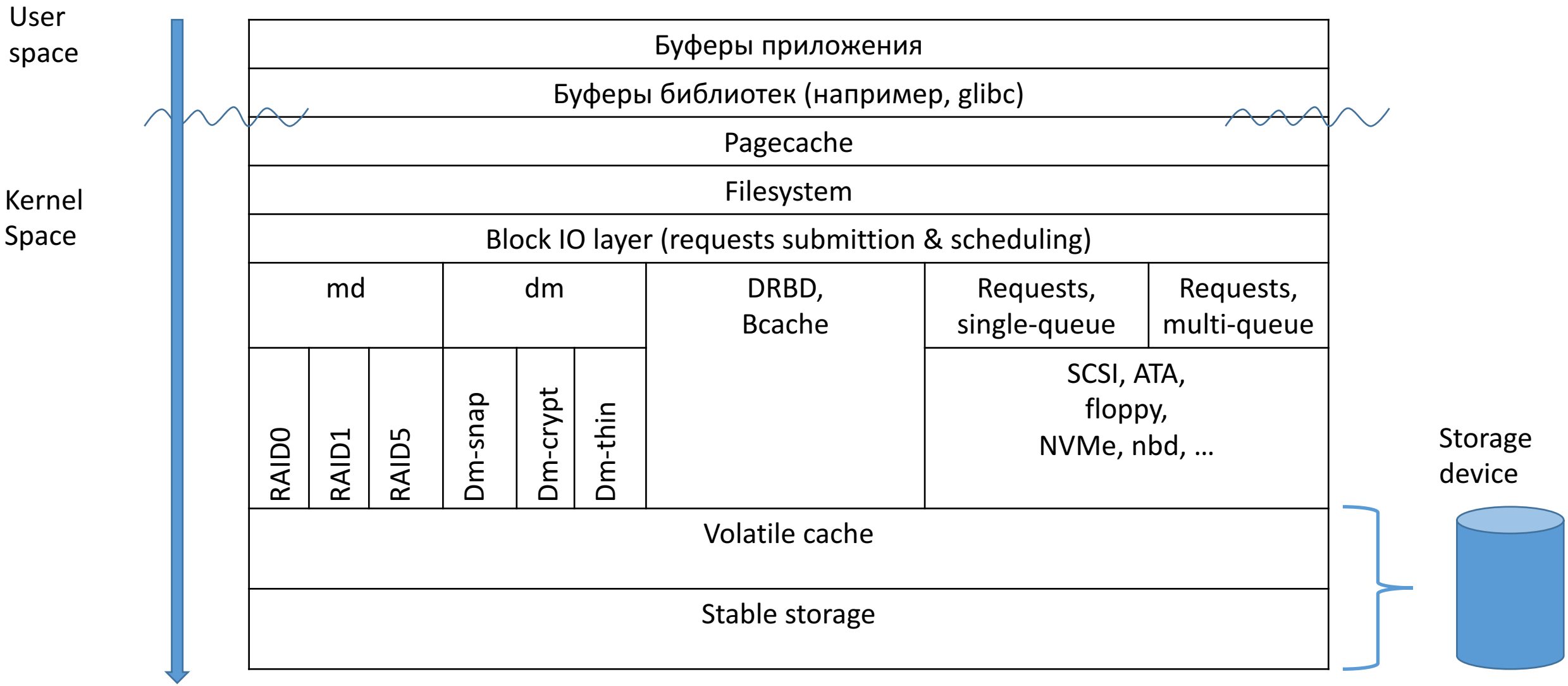
Ещё одна проблема RAID5

Восстановление данных занимает достаточно долго времени*, притом в течение всего этого промежутка на оставшиеся диски создаётся высокая нагрузка, что повышает вероятность выхода из строя ещё одного диска во время перестроения RAID5.

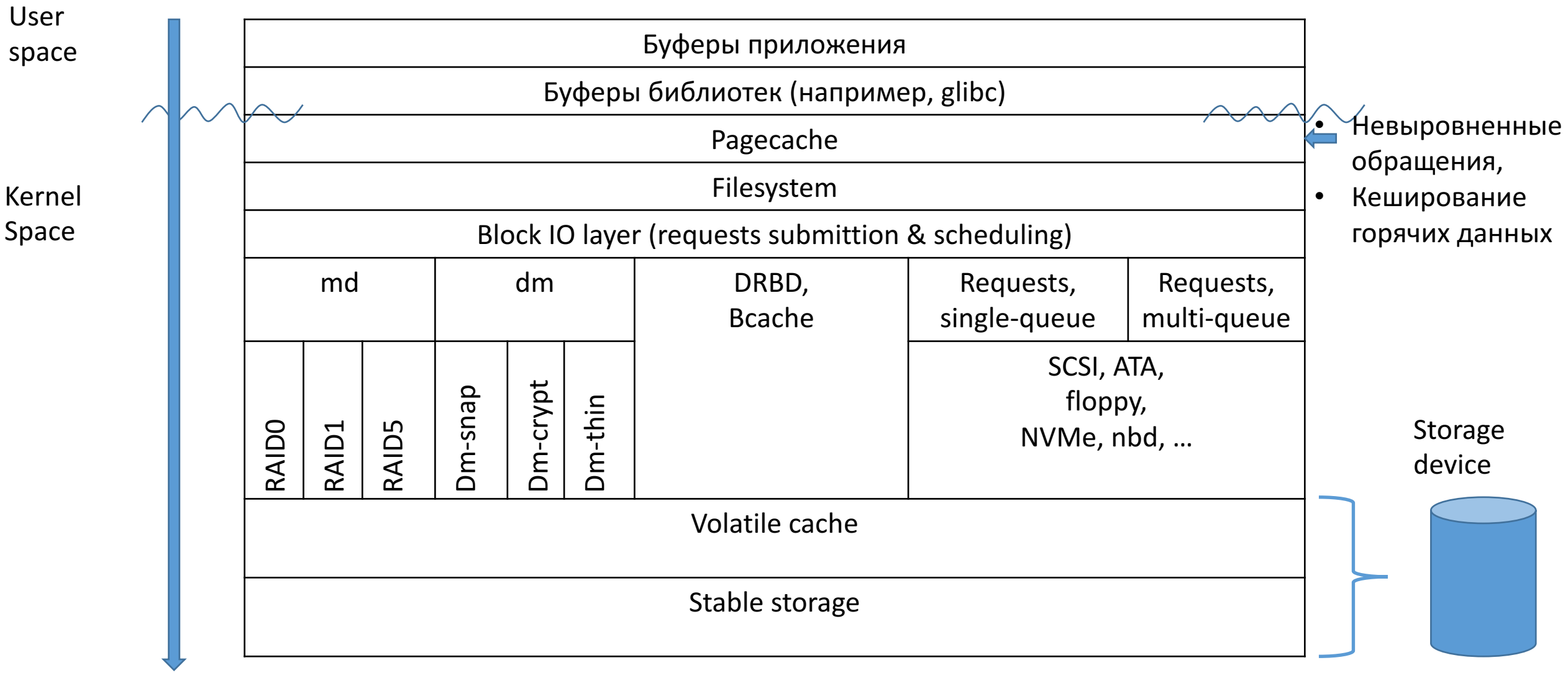
В практике наблюдалось достаточно количество ситуаций, когда во время перестроения массива отказывал второй диск. По этой причине на больших массивах от RAID5 отказались.

** Перезаписать диск 10Tb на скорости 100Mb/sec займёт порядка полутора суток.*

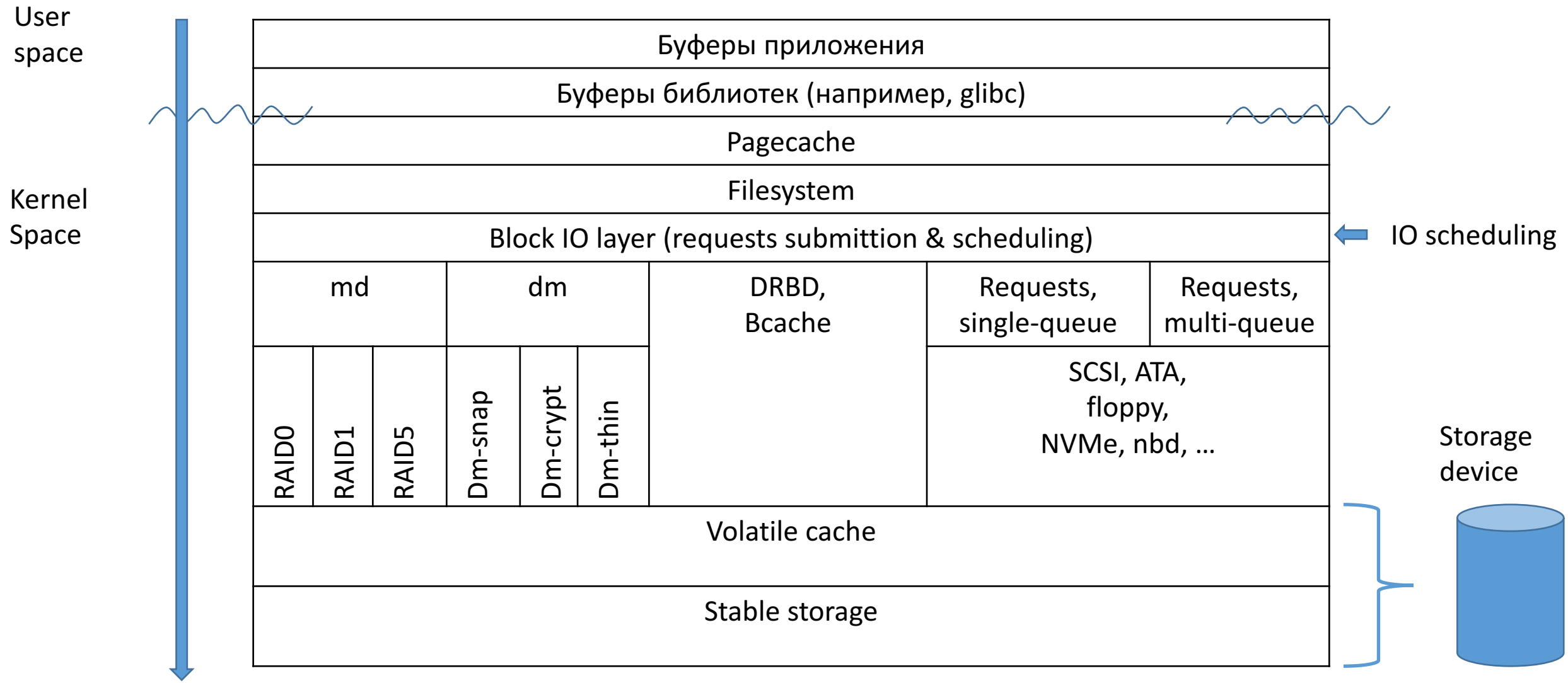
Путь данных от приложения до диска (обзорно)



Путь данных от приложения до диска (обзорно)



Путь данных от приложения до диска (обзорно)



Отступление: deadlock’и в условиях ограниченных ресурсов

Block IO layer не имеет права выделять память, поскольку он обычно используется в ситуациях, когда памяти уже мало и изменённые страницы надо сбрасывать на диск.

Поэтому все struct bio выделяются из пула фиксированной длины.

Представим себе ситуацию:

- Есть mempool, состоящий из 16 элементов,
- 16 потоков делают чтения по 8Kb из RAID10-массива,
- Слой RAID1 должен сделать запросы к обеим частям зеркала, чтобы сравнить их. Он выделяет 16 экземпляров bio.
- Слой RAID0 должен разрезать запросы на 4Kb запросы к разным частям страйпов. Память под них взять неоткуда.

Отступление: deadlock’и в условиях ограниченных ресурсов

Block IO layer не имеет права выделять память, поскольку он обычно используется в ситуациях, когда памяти уже мало и изменённые страницы надо сбрасывать на диск.

Поэтому все struct bio выделяются из пула фиксированной длины.

Представим себе ситуацию:

- Есть mempool, состоящий из 16 элементов,
- 16 потоков делают чтения по 8Kb из RAID10-массива,
- Слой RAID1 должен сделать запросы к обеим частям зеркала, чтобы сравнить их. Он выделяет 16 экземпляров bio.
- Слой RAID0 должен разрезать запросы на 4Kb запросы к разным частям страйпов. Память под них взять неоткуда.

Решение: bio, соответствующие более вложенным устройствам, обрабатываются перед bio от более высокоуровневых устройств.

Упражнение: сравните со схемой, где на блокировках вводится частичный порядок и блокировки берутся только в порядке возрастания.

Путь данных от приложения до диска (обзорно)