**Программный комплекс хранения данных, с настраиваемой архитектурой - ориентированный на использование в программно-аппаратных комплексах для хранения данных (Node Appliance)**

***Реестровая запись №5850 от 20.09.2019 - СХД АРГО***

**Состав программного комплекса хранения:** ISCRA OS (Intellectual Scalable Clusterised Appliance) (OpenSolaris Base); ZFS; High Availability Framework; Cluster Framework; Web; Monitorin.

**Node Appliance** — это готовый к эксплуатации, протестированный и оптимизированный узел, с полной совместимостью программного комплекса хранения данных STORAGE by ARGO.TECH и оборудования.

**Состав (Node Appliance):**
Надёжное оборудование (внесено в Реестр Минпромторг);
Высокопроизводительные накопители (HDD / SSD / NVMe);
Предустановленный, оптимизированный программный комплекс хранения данных.

**STORAGE обладает следующими функциональными возможностями:**

| № | Возможность | Описание | Ценность |
|---|---|---|---|
| 1 | **Пулы (pools)** | Объединение дисков в логические группы с динамическим распределением места | Простое масштабирование и высокая гибкость без переразметки томов |
| 2 | **Настраиваемый размер блока** | Нет единого размера блока файловой системы | Возможно настраивать исходя из профиля нагрузки для разных датасетов |
| 3 | **Copy-on-write + checksumming** | Запись без перезаписи и контроль целостности каждого блока | консистентность и автоматическое исправление ошибок |
| 4 | **Снимки (snapshots)** | Мгновенные консистентные копии файловых систем | Безопасное обновление, резервное копирование, откат изменений |
| 5 | **Клоны** | Writable-копии снимков без дублирования данных | Быстрое создание тестовых или dev-сред без затрат места |
| 6 | **Репликация** | Асинхронная и инкрементальная передача данных между узлами | Простое построение DR и резервных площадок |
| 7 | **ARC / L2ARC кэш** | Интеллектуальное кэширование в RAM и SSD | Ускорение чтения и оптимизация “горячих” данных |
| 8 | **Журнал намерений** | Журналирование синхронных операций на SSD/NVRAM | Ускорение записи для баз данных |
| 9 | **Сжатие и дедупликация** | LZ4/ZSTD компрессия и хэш-дедупликация | Снижение затрат на дисковое пространство |
| 10 | **Thin provisioning** | Выделение логического объёма больше физического | Оптимизация использования хранилища |
| 11 | **Квоты и резервации** | Контроль доступного объёма на уровне dataset или пользователя | Прозрачное управление ресурсами |
| 12 | **NFS / CIFS/SMB / iSCSI / FC** | Мультипротокольный доступ к данным | Единая СХД для разных платформ и клиентов |
| 13 | **COMSTAR target framework** | Встроенный FC и iSCSI таргет | Бесшовная интеграция с виртуализацией и SAN |
| 14 | **Failover-кластер (HA)** | Реплицированные пулы с автоматическим переключением | Непрерывная доступность сервисов |
| 15 | **Мониторинг и FMA** | Fault Management Architecture, | Предупреждение сбоев до их |

| | | | |
|---|---|---|---|
| | | SNMP, Prometheus | возникновения |
| 16 | **Шифрование dataset** | Встроенное AES-шифрование на уровне файловой системы | Защита данных при хранении и передаче |
| 17 | **ACL / AD / LDAP интеграция** | Аутентификация пользователей и контроль прав | Централизованное управление безопасностью |
| 18 | **Веб- и CLI-управление** | Полное администрирование через GUI, CLI и API | Простота внедрения и автоматизации |
| 19 | **Интеграция с различными системами виртуализации** | Поддержка NFS/iSCSI datastore и vVOL | Оптимизированная работа виртуальных сред |
| 20 | **Self-healing и scrub** | Автоматическая проверка и исправление ошибок | Минимизация риска потери данных |
| 21 | **Scale-up расширение** | Добавление дисков без простоя | Линейный рост ёмкости и производительности |
| 22 | **WORM / Immutable dataset** | Режим неизменяемых данных | Соответствие требованиям compliance и архивов |
| 23 | **Полная телеметрия (arcstat, zpool, iostat)** | Расширенная аналитика состояния пула и I/O | Прозрачность производительности и предсказуемость SLA |

**High Availability: (высокая доступность):** Комплекс мер, направленный на обеспечение функционирования пула хранения, а так же связанных с этим пулом сервисов, при выходе из строя контроллера (контроллерной ноды), являющейся "владельцем" пула в текущий момент времени, либо необходимого минимума путей от ноды-владельца до дисковых ресурсов, являющихся частью защищаемого пула хранения и необходимых для его функционирования.

**Sync/Async Replication:** Асинхронная репликация в STORAGE применяется в качестве катастрофо-устойчивого решения для хранения реплик томов или датасетов (наборов данных) на других СХД на удаленных площадках. Репликацию можно выполнять на относительно нестабильных каналах связи, или когда пиковая нагрузка выше пропускной способности канала связи.

Для работы функционала асинхронной репликации требуется совместимая СХД.
Репликация производится на уровне блоков, каждую итерацию передачи и записи подвергаются только измененные блоки — это позволяет экономить сетевой трафик и время, требующееся на синхронизацию.

*Помимо этого, трафик, который передается по сети связи - так же возможно дополнительно сжать.*

Так же предусмотрена возможность лимитирования полосы пропускания на передающей стороне - для возможности некоторой балансировки доступной полосы между разными инстансами сервиса или снижения интенсивности считывания данных с пула хранения.

Сама же полезная нагрузка передается в зашифрованном канале связи. По умолчанию используется шифр **aes128-gcm**, который реализуется с приемлемой производительностью многими современными процессорами.

**RAID / Erasure Coding:** в STORAGE используется алгоритм **erasure coding** с двойной или тройной защитой блока данных.

После подтвержденной записи блока данных этот блок практически невозможно потерять или испортить, его целостность проверяется и гарантируется соответственно двойной или тройной контрольной суммой. Технологии ARGO.TECH выполняют восстановление целостности массива за кратчайшее время, даже в случае одновременного отказа нескольких дисков.

STORAGE распределяет блоки данных по всем доступным накопителям и контроллерам, обеспечивая высокий параллелизм, молниеносное время отклика, и высокую производительность. Решение позволяет с одинаково высокой эффективностью обслуживать как традиционные нагрузки СУБД (OLTP), так и задачи, связанные с обработкой десятков и сотен петабайт неструктурированных данных.

**Multipathing:** (многопутевой ввод-вывод) — это технология, которая использует несколько физических путей для подключения к одному устройству хранения данных. Её основная цель — повышение отказоустойчивости (при отказе одного пути система продолжает работу по-другому) и производительности (за счет распределения нагрузки между несколькими путями). Реализуется это с помощью специального программного обеспечения, которое объединяет несколько физических соединений в одно логическое устройство.

В STORAGE это реализовано следующим образом: В многоконтроллерной СХД логический том “принадлежит” одному из контроллеров, хотя доступен через все контроллеры.

Сервис ALUA определяет, какие пути к тому являются предпочтительными, а какие — альтернативными:

**Оптимальный путь (Active/Optimized):** проходит через контроллер, который “владеет” томом. Обеспечивает максимальную производительность, так как данные не требуют дополнительной передачи между контроллерами.

**Неоптимальный путь (Active/Non-Optimized):** проходит через “не владеющий” контроллер. Такой путь полностью работоспособен, но может иметь несколько сниженную производительность из-за необходимости передачи данных между контроллерами.

***Вот один из примеров на практике:***
*1. Обнаружение путей: при подключении клиента к СХД многопутевое программное обеспечение (multipath) обнаруживает все доступные пути к каждому тому.*

*2. Статус состояний: СХД сообщает информацию о состоянии каждого пути (оптимальный или неоптимальный).*

*3. Маршрутизация I/O: Драйвер multipath приоритизирует трафик через оптимальные пути, но сохраняет информацию о всех доступных путях.*

*4. Автоматическое переключение: при отказе контроллера или соединения система автоматически переключает весь ввод-вывод на оставшиеся пути.*

STORAGE использует **журнал намерений** для ускорения синхронной записи, который находится на быстром выделенном устройстве хранения (SSD), которое хранит этот журнал вместо дисков основного пула. Такое разделение значительно повышает производительность записи для приложений, использующих синхронную запись, таких как: базы данных и виртуальные машины, перенося процесс журналирования на более быстрый носитель.

***Восстановление после сбоя:***
*Если система внезапно выключается, при том, используя информацию из журнала, она повторно применяет все записанные изменения к файлам данных, гарантируя целостность и согласованность.*

***Надёжность:*** *гарантирует, что даже при внезапном сбое данные останутся в согласованном состоянии.*

***Атомарность и устойчивость:*** *поддерживает ACID-свойства баз данных.*

***Производительность:*** *позволяет записывать данные в более произвольном порядке, а не в строгом соответствии с порядком транзакций, что может повысить производительность.*

**Self-Healing:** В STORAGE реализована полноценная система управления авариями, сбоями и оповещениями. На верхнем уровне стек управления неисправностями включает в себя детекторы ошибок и наблюдений, механизм диагностики и агенты реагирования.

***Детекторы ошибок*** *обнаруживают ошибки в системе и немедленно выполняют необходимые действия. Детектор ошибок выдаёт чётко определённый отчёт в диагностическую систему.*

***Детекторы наблюдения*** *сообщают о состояниях в системе, которые не являются симптомами неисправностей или дефектов. Детектор наблюдения выдаёт чётко определённый информационный отчёт, который может быть передан в диагностическую систему или просто зарегистрирован.*

***Модуль диагностики*** *интерпретирует отчеты и определяет, следует ли диагностировать неисправность, дефект или предупреждение. После принятия такого решения модуль диагностики выдает список подозреваемых, описывающий ресурс или набор ресурсов, которые могут быть причиной проблемы или состояния. Ресурс может быть немедленно выведен из эксплуатации для устранения проблемы или замены.*
*Если список "подозреваемых" включает несколько подозреваемых (например, если модуль диагностики не может изолировать одного подозреваемого), каждому подозреваемому присваивается вероятность того, что он является ключевым подозреваемым. Сумма вероятностей в этом списке составляет 100%. Списки подозреваемых интерпретируются агентами реагирования.*

***Агенты реагирования*** *предпринимают действия на основе списка подозрительных проблем. Реакции включают в себя регистрацию сообщений в журнале, отключение потоков ЦП, освобождение страниц памяти и освобождение устройств ввода-вывода.*

**Health Monitoring/Performance monitor:** В текущей реализации STORAGE, на каждой ноде устанавливается компонент, который осуществляет сбор метрик (**CPU, RAM, NET, IO, ZFS, SMF, IPMI и т.д.**).

Опрос метрик осуществляется компонентом, который развернут и запущен в изолированном окружении. Хранение собранных метрик осуществляется в базу, которая располагается в том же окружении. В качестве бэкенда хранилища используется выделенный датасет в системном пуле.

Помимо этого, компонент опрашивает **exporter** соседней ноды кластера, таким образом обеспечивается избыточность сохраненной информации (на случай выхода из строя дисков, ноды или программных компонентов). За визуализацию отвечает grafana, который развернут и запущен в отдельном изолированном окружении.

*Сервисы расширенного мониторинга метрик являются дополнительным компонентом, который может быть как включен, так и не включен в состав лицензии (права на использование) приобретаемой СХД.*

**ARC (Adaptive Replacement Cache)** — адаптивный алгоритм кэширования данных, используемый в STORAGE, который оптимизирует производительность, отслеживая как недавно, так и часто используемые данные. Он хранит данные в оперативной памяти (ОЗУ), чтобы ускорить доступ к ним, и отличается от простых алгоритмов, таких как LRU и MFU тем, что учитывает историю вытеснения данных, обеспечивая более эффективное управление кэшем.

*За счет эффективного управления данными в кэше, ARC может значительно ускорить доступ к часто используемым данным, снижая нагрузку на более медленные диски.*

**L2ARC (Level 2 ARC):** это кэш адаптивной замены второго уровня. Данный кэш располагается на высокопроизводительных дисковых устройствах, SSD или NVMe накопителях. Когда диски кэша присутствуют в пуле дисков, они кэшируют часто используемые данные, которые не помещаются в ARC.

***L2ARC**, является вторым кэшем чтения. **L2ARC** перехватывает выпадающие из ARC элементы. При применении быстрого диска с большим резервом перезаписи для кэширования данных ARC, вы можете одновременно уменьшить нагрузку чтения в вашем основном хранилище системы и улучшить производительность чтения.*

**I/O Prioritization / Scheduling:** Приоритезация и планирование ввода-вывода (I/O) используется для управления и упорядочивания запросов ввода-вывода к устройствам хранения данных. Она направлена на балансировку пропускной способности и скорости отклика системы путем назначения приоритетов различным запросам, обеспечивая критически важным процессам более быстрый доступ к ресурсам ввода-вывода.

*Это достигается различными методами, такими как объединение запросов, сортировка запросов для минимизации времени поиска и предоставление определённым запросам приоритета на основе таких факторов, как уровни приоритета или сроки выполнения.*

**Compression:** Сжатие в STORAGE совершенно незаметно для пользователя. Хотя сжатие увеличивает потребление ресурсов, пользователи не должны заметить изменения в скорости хранения. **Важно отметить, что сами файлы не сжимаются.** Сжатие сжимает данные на основе записи.

*При сохранении файла сам файл не сжимается, а изначально сжимается запись перед сохранением. Если файловая система не может сжать файл, он сохраняется в несжатом виде, чтобы избежать потери данных. Степень сжатия устанавливается с помощью различных алгоритмов.*

*STORAGE поддерживает несколько алгоритмов сжатия, каждый из которых имеет свои компромиссы с точки зрения степени сжатия и вычислительных затрат. Некоторые из распространённых алгоритмов сжатия включают:*

***LZ4:** Быстрый и простой алгоритм сжатия, обеспечивающий хорошую степень сжатия при минимизации нагрузки на процессор.*

***Gzip:** широко используемый алгоритм сжатия, обеспечивающий более высокую степень сжатия, но за счёт увеличения загрузки процессора во время сжатия и распаковки.*

***Zstd:** Современный и универсальный алгоритм сжатия, обеспечивающий широкий диапазон степеней сжатия и скоростей, подходящий для различных вариантов использования.*

***LZJB:** Алгоритм сжатия по умолчанию, используемый в файловой системе, обеспечивающий баланс между эффективностью сжатия и скоростью.*

***ZIE:** кодирование нулевой длины полезно для наборов данных с большими блоками нулей*

**Deduplication:** позволяет сократить место на дисках, используемое для хранения данных. Блоки перед записью проверяются на уникальность и на диск записываются только уникальные блоки. Информация о блоках хранится в **справочной таблице,** которая связывает файлы и данные пула с фактическими блоками хранения, содержащими эти данные.

*Таблица дедупликации — это часть метаданных или пула. Таблица дедупликации может временно храниться в кэше для повышения скорости и обеспечения многократного использования, но таблица дедупликации не является дисковым кэшем. Пул может содержать любую сосуществующую смесь дедуплицированных и недедуплицированных данных.*

*Расчет дедупликации можно наглядно представить в виде таблицы. Ниже приведены два примера: для резервного копирования и для файлового хранилища.*

*Этот пример показывает, как дедупликация сокращает объем данных при регулярном резервном копировании, когда большая часть данных не меняется.*

| Показатель | Значение | Описание |
|---|---|---|
| Общий объем данных до | 300 ТБ | Ежедневное полное копирование 10 ТБ данных в |

| | | |
|---|---|---|
| дедупликации | | течение 30 дней. |
| Объем данных после дедупликации | 45 ТБ | Объем уникальных блоков данных, которые сохранились после дедупликации. |
| Коэффициент дедупликации | 6,67:1 | Рассчитывается как (300ТБ/45ТБ). На каждые 6,67 ТБ исходных данных требуется 1 ТБ хранилища. |
| Экономия места | 85% | Рассчитывается как (1-(45ТБ}/300)). Экономия 255 ТБ места. |

*Расчет дедупликации для файлового хранилища. Этот пример демонстрирует, как дедупликация работает с повторяющимися файлами в корпоративной среде, например, с документами и отчетами.*

| Показатель | Значение | Описание |
|---|---|---|
| Исходный объем данных | 50 ГБ | Суммарный размер 500 копий документа объемом 100 МБ. |
| Объем данных после дедупликации | 100 МБ | Размер одной уникальной копии документа, хранящейся на сервере. |
| Коэффициент дедупликации | **500:1** | Рассчитывается как (50ГБ/100МБ). На каждые 500 ГБ исходных данных требуется 100 МБ хранилища. |
| Экономия места | **99,8%** | Рассчитывается как (1-(100МБ/50ГБ)). |

**Parallel Scrubbing and Resilvering: Scrubbing** (проверка целостности) — это процесс регулярного сканирования дисков для поиска и исправления ошибок. **Resilvering** (восстановление) — это процесс восстановления данных с использованием избыточной информации после замены вышедшего из строя диска. Использование параллельных версий этих процессов значительно ускоряет их выполнение.

**Pooled Storage Architecture:** Пул хранения представляет из себя набор устройств, которые предоставляют свои ресурсы для хранения структуры, метаданных, полезных данных.

*Все наборы данных (датасеты - dataset), размещенных в пуле хранения, используют эти ресурсы сообща.*

**RAID:** В STORAGE используется несколько типов RAID

**Mirror - аналог raid-1**

Используется при создании пула, с характеристиками, аналогичными raid-1 или raid-10. Каждое зеркало может состоять из 2 и более устройств. Данные, записываемые на эти устройства - реплицируются между всеми членами "зеркала". В отличии от устройств ниже - для зеркал не вычисляется Parity.

**Raidz - аналог raid-5**

Используется при создании пула, с характеристиками, аналогичными raid-5 или raid-50. Может состоять из 3 и более устройств. Данные и Parity, записываемые на эти устройства - распределяются между всеми членами группы (нет четко выраженных дисков с данными или с Parity). Каждая такая группа выдерживает отказ 1 устройства (диска).

**Raidz2 - аналог raid-6**

Используется при создании пула, с характеристиками, аналогичными raid-6 или raid-60. Может состоять из 4 и более устройств. Данные и Parity, записываемые на эти устройства - распределяются между всеми членами группы (нет четко выраженных дисков с данными или с Parity). Каждая такая группа выдерживает отказ 2 устройств (дисков).

**Raidz3 - аналог raid с чередованием блоков с тройным распределением четности**

Каждая такая группа выдерживает отказ 3 устройств (диска).

*Группы являются усовершенствованной реализацией классических RAID-массивов. Главное отличие заключается в устранении проблемы "дыры записи" (write hole), характерной для традиционных RAID5 и RAID6, что обеспечивает более надёжную защиту данных.*

**Основные отличия от обычных RAID**

| Критерий | RAIDZ | Обычные RAID (например, RAID5) |
|---|---|---|
| Распределение данных | Динамическое чередование данных и информации о чётности (parity) происходит по всем дискам в группе. Блоки данных и блоки чётности записываются одновременно. | Чередование данных и чётности выполняется блоками фиксированного размера. Это может приводить к несогласованности данных и чётности после сбоя питания — "дыре записи". |
| Целостность данных | Устраняет проблему "дыры записи" благодаря механизму "копирование-при-записи" (Copy-on-Write, COW). Когда блок данных перезаписывается, файловая система записывает новый блок на новое место, а не поверх старого, обновляя затем метаданные. Это обеспечивает атомарность операций и предотвращает несогласованность. | Старые реализации RAID могут быть подвержены проблеме "дыры записи", когда сбой электропитания во время операции записи приводит к потере данных или неверной чётности. |
| Гибкость | Не требует, чтобы все диски в группе имели одинаковый размер. Пространство на дисках разного объёма используется более эффективно, что нехарактерно для традиционных RAID. | Как правило, требует использования дисков одинакового размера. Ёмкость массива ограничивается размером самого маленького диска, и лишнее пространство на более крупных дисках не используется. |
| Расширяемость | Расширение группы путём добавления одного диска невозможно, так как это изменило бы структуру чередования. Вместо этого нужно добавить новую группу дисков к пулу хранения. | Некоторые контроллеры RAID могут позволять добавление дисков, но это часто требует сложных операций и может быть рискованным. |
| Избыточность и восстановление | Хранит информацию о чётности, используя контрольные суммы для проверки целостности данных. В случае обнаружения повреждённых данных система может восстановить их, используя информацию о чётности. | Использует чётность для восстановления данных, но не имеет такого механизма самовосстановления и проверки целостности |

**Thin Provisioning:** позволяет выделять пространство хранилища виртуально, без немедленного выделения физического пространства. При создании нового тома ему назначается ограничение по размеру, но он использует физическое хранилище только по мере фактической записи данных в том. Это обеспечивает более гибкое и эффективное использование ресурсов хранилища.

**Снимки (Snapshots)**

Принцип работы: когда данные изменяются, STORAGE записывает новую версию в новый блок, не перезаписывая старые. Снимок фиксирует состояние файловой системы на определенный момент времени, просто создавая ссылку на существующие блоки.

***Использование:*** *Идеальны для создания резервных копий или для сохранения состояния файловой системы до проведения изменений. В случае ошибки можно легко откатить изменения, вернувшись к состоянию снимка.*

***Создание:*** *создаются быстро, часто рекурсивно (все дочерние файловые системы одновременно).*

***Местоположение:*** *Снимки хранятся в том же пуле устройств хранения данных, что и исходный набор данных, и не требуют выделения отдельного дискового пространства.*

***Доступ:*** *Снимки можно просматривать и восстанавливать из них данные.*

**Клоны (Clones)**

Принцип работы: Клон — это полностью новая, "живая" файловая система (или том), которая изначально имеет то же содержимое, что и снимок, на основе которого она была создана.

***Использование:*** *Полезны для создания изолированных сред для разработки, тестирования или для временных пользовательских каталогов, не копируя при этом все данные.*

***Совместное использование данных:*** *Клоны и их исходный снимок (а также исходный набор данных) совместно используют нетронутые блоки данных, поэтому первоначальное создание клона занимает минимум пространства.*

***Расширение:*** *Исходный набор данных и снимок сохраняют свое состояние, тогда как на клоне можно производить любые изменения, создавая для него свои собственные блоки данных.*

**Ключевые различия: снапшот vs клон**

| Характеристика | Снапшот (Snapshot) | Клон (Clone) |
|---|---|---|
| **Доступ** | Только для чтения. | Доступен для чтения и записи. |
| **Создание** | Создаётся из активной файловой системы или тома. | Всегда создаётся из существующего снапшота. |
| **Начальный размер** | Не занимает места (делит блоки с источником). | Не занимает места (делит блоки с родительским снапшотом). |
| **Использование места** | Занимает место только по мере изменения исходных данных. | Занимает место по мере изменения клонированных данных. |
| **Цель** | Хранение "точки во времени", резервное копирование, откат. | Создание отдельной, изменяемой копии для тестирования или других задач. |
| **Удаление** | Можно удалить, если на него не ссылаются клоны. | Можно удалить в любое время. Перед удалением родительского снапшота, клон должен быть удалён. |

*Ограничений на количество снапшотов и клонов с точки зрения жёстко заданных лимитов, как правило, нет. Теоретические пределы, заложенные в архитектуру файловой системы, очень велики ($2^{64}$) .*

**Главное "ограничение" на количество снапшотов и клонов** — это не жёсткий лимит, а управляемость и потребление системных ресурсов (памяти и дискового пространства). Тысячи и даже десятки тысяч снапшотов в одном пуле возможны при наличии аппаратных ресурсов.

**Replication:** позволяет создавать и передавать потоковые представления данных (снапшотов) из одного пула в другой. Этот механизм используется для резервного копирования, миграции и аварийного восстановления.

**Типы репликации**

**Полная репликация:** при первой передаче отправляется полный снимок файловой системы. Это может занять много времени, так как копируются все данные.

**Инкрементальная репликация:** после первой полной репликации можно передавать только те блоки данных, которые изменились между двумя последовательными снимками. Это делает последующие операции быстрыми и экономит ресурсы.

***Пример***
*Резервное копирование и аварийное восстановление (Disaster Recovery)*
*Это один из самых распространённых сценариев использования. Репликация позволяет поддерживать актуальную копию данных на удалённой системе. В случае сбоя основной системы, вы можете быстро переключиться на резервную.*

**Export/Import Pools** Процедуры экспорта и импорта пулов STORAGE необходимы для переноса пулов с одного сервера на другой. Например, при замене оборудования, переносе пула в другую операционную систему или для восстановления данных.

**Native Encryption** в STORAGE встроенная в файловую систему функция, которая позволяет шифровать данные непосредственно на уровне файловой системы. Это обеспечивает защиту данных от несанкционированного доступа, а шифрование выполняется с использованием алгоритма **AES (Advanced Encryption Standard)**.

**Dataset hierarchy:** Иерархия наборов данных (dataset hierarchy) древовидная структура, которая организует все данные в пуле хранения. Она обеспечивает гибкое и гранулированное управление хранилищем: свойства наследуются от родительских наборов данных к дочерним.

**Quota (Квота):** Квота устанавливает максимальный объём дискового пространства, который может использовать датасет и все его потомки (вложенные датасеты, снимки и клоны). Если квота установлена, попытки записать данные, которые превышают этот лимит, будут завершаться ошибкой.

**Reservation (Резервирование):** Резервирование гарантирует, что определённое количество дискового пространства из пула всегда будет доступно для данного датасета. Это минимальный гарантированный объём. Зарезервированное пространство не может быть использовано другими датасетами, даже если оно пока не занято самим датасетом.

**Сравнение Quota и Reservation**

| Критерий | Quota | Reservation |
|---|---|---|
| Функция | Устанавливает верхний предел использования пространства. | Гарантирует минимальный объём пространства. |
| Влияние на родителя | Влияет на доступное пространство родительского датасета, так как учитывается при подсчёте занятого места. | Учитывается в занятом пространстве родительского датасета, уменьшая доступное для других датасетов место. |
| Управление | Предотвращает "захват" всего свободного места одним датасетом. | Защищает важные датасеты от нехватки места, если пул будет заполняться другими. |
| Поведение при нехватке места | Запись новых данных, превышающих лимит, блокируется. | Установка резервации невозможна, если в пуле нет достаточного свободного места. |
| Совместное использование | Можно использовать оба механизма одновременно, чтобы и ограничивать, и гарантировать место. | Можно использовать оба механизма одновременно. |

**Transactional Semantics / Consistency Groups:** транзакционная семантика и группы согласованности в

STORAGE — это механизмы, обеспечивающие целостность данных и защиту от повреждений. Вместо традиционного подхода, где данные перезаписываются, используется модель "копирование при записи" (Copy-on-Write). Транзакционный механизм Copy-on-Write (CoW): файловой системы всегда записывает новые данные в свободное место. Это врожденно асинхронный процесс, так как файловой системе не нужно перезаписывать существующие блоки. Вместо этого она создает новые, а старые данные остаются доступными до тех пор, пока на них есть ссылки.

**Multithreading and async I/O**

STORAGE использует как многопоточность, так и асинхронный ввод-вывод для эффективной обработки операций с диском. Эти механизмы встроены в ядро файловой системы. Многопоточность и асинхронный ввод-вывод — это подходы к выполнению задач, которые не блокируют работу программы.