Лекция №14

Файловые системы SAN, GFS, DFS

SCSI

- *Small Computer System Interface* - инт… объ… ных по сво… к как жёс… маг… CD, DV… и т. д.
- Существует реализация системы команд SCSI поверх оборудования (контроллеров и кабелей) IDE/ATA/SATA, называемая ATAPI - ATA Packet Interface.

Наименование	Разрядность шины	Частота шины	Пропускная способность	Максимальная длина кабеля	Максимальное количество устройств
SCSI	8 бит	5 МГц	5 МБайт/сек	6 м	8
Fast SCSI	8 бит	10 МГц	10 МБайт/сек	1,5-3 м	8
Wide SCSI	16 бит	10 МГц	20 МБайт/сек	1,5-3 м	16
Ultra SCSI	8 бит	20 МГц	20 МБайт/сек	1,5-3 м	5-9
Ultra Wide SCSI	16 бит	20 МГц	40 МБайт/сек	1,5-3 м	5-8
Ultra2 SCSI	8 бит	40 МГц	40 МБайт/сек	12 м	8
Ultra2 Wide SCSI	16 бит	40 МГц	80 МБайт/сек	12 м	16
Ultra3 SCSI	16 бит	40 МГц DDR	160 МБайт/сек	12 м	16
Ultra-320 SCSI	16 бит	80 МГц DDR	320 МБайт/сек	12 м	16
Ultra-640 SCSI	16 бит	160 МГц DDR	640 МБайт/сек		

SAS (**Serial Attached SCSI**)

- компьютерный интерфейс, разработанный для обмена данными с такими устройствами, как жёсткие диски, накопители на оптическом диске и т. д.
- использует последовательный интерфейс для работы с непосредственно подключаемыми накопителями.
- разработан для замены параллельного интерфейса SCSI и позволяет достичь более высокой пропускной способности, чем SCSI
- совместим с интерфейсом SATA.

SCSI vs SAS

- SAS использует последовательный протокол передачи данных между несколькими устройствами, и, таким образом, использует меньшее количество сигнальных линий.
- Интерфейс SCSI использует общую шину. Таким образом, все устройства подключены к одной шине, и с контроллером одновременно может работать только одно устройство. Интерфейс SAS использует соединения точка-точка — каждое устройство соединено с контроллером выделенным каналом.
- В отличие от SCSI, SAS не нуждается в терминации шины пользователем.
- В SCSI имеется проблема, связанная с тем, что скорость передачи информации по разным линиям, составляющим параллельный интерфейс, может отличаться. Интерфейс SAS лишён этого недостатка.
- SAS поддерживает большое количество устройств (> 16384), в то время как интерфейс SCSI поддерживает 8, 16, или 32 устройства на шине.
- SAS поддерживает высокие скорости передачи данных (1,5, 3,0 или 6,0 Гбит/с). Такая скорость может быть достигнута при передаче информации на каждом соединении инициатор-целевое устройство, в то время как на шине SCSI пропускная способность шины разделена между всеми подключёнными к ней устройствами.
- SAS поддерживает подключение устройств с интерфейсом.
- SAS использует команды SCSI для управления и обмена данными с целевыми устройствами.

NAS (Network Attached Storage)

- сетевая система хранения данных, сетевое хранилище.
- представляет собой одно устройство с некоторым дисковым массивом, подключенный к сети (обычно локальной) и поддерживающий работу по принятым в ней протоколам. Часто диски в NAS объединены в RAID массив. Несколько таких устройств могут быть объединены в одну систему.
- Обеспечивает надёжность хранения данных, лёгкость доступа для многих пользователей, лёгкость администрирования, масштабируемость.

SAN (*Storage Area Network)*

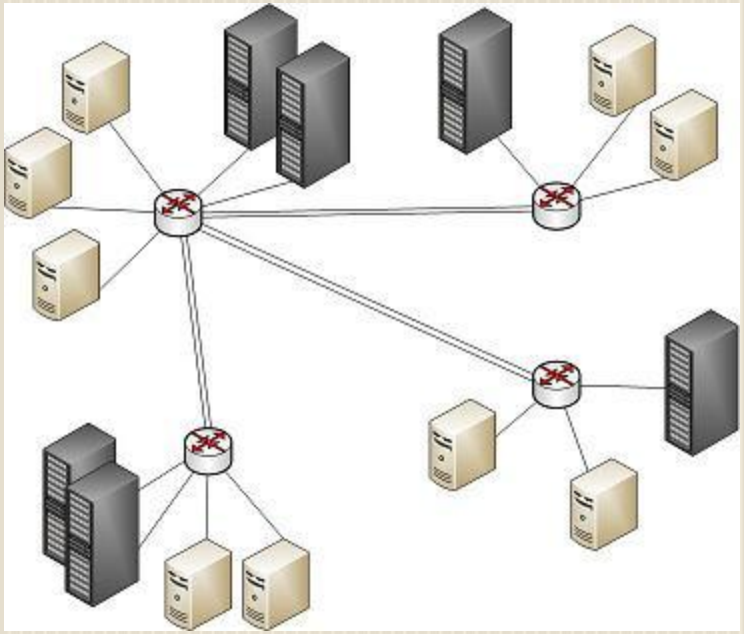
- представляет собой архитектурное решение для подключения внешних устройств хранения данных, таких как дисковые массивы, ленточные библиотеки, оптические накопители к серверам таким образом, чтобы операционная система распознала подключённые ресурсы, как локальные.
- Не путать с NAS!!!

SAN vs NAS

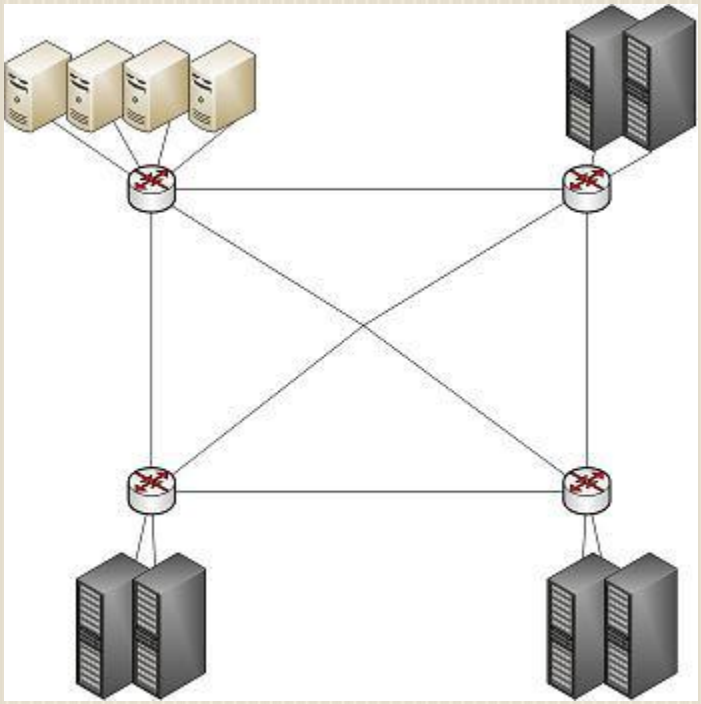
- Кто обслуживает файловую систему?
- NAS – клиент оперирует понятием файл
- SAN – блочное устройство, передача данных осуществляется на уровне SCSI-блоков
- Обслуживание файловой системы в SAN возложено на клиентский компьютер, т.н. Raw-устройство

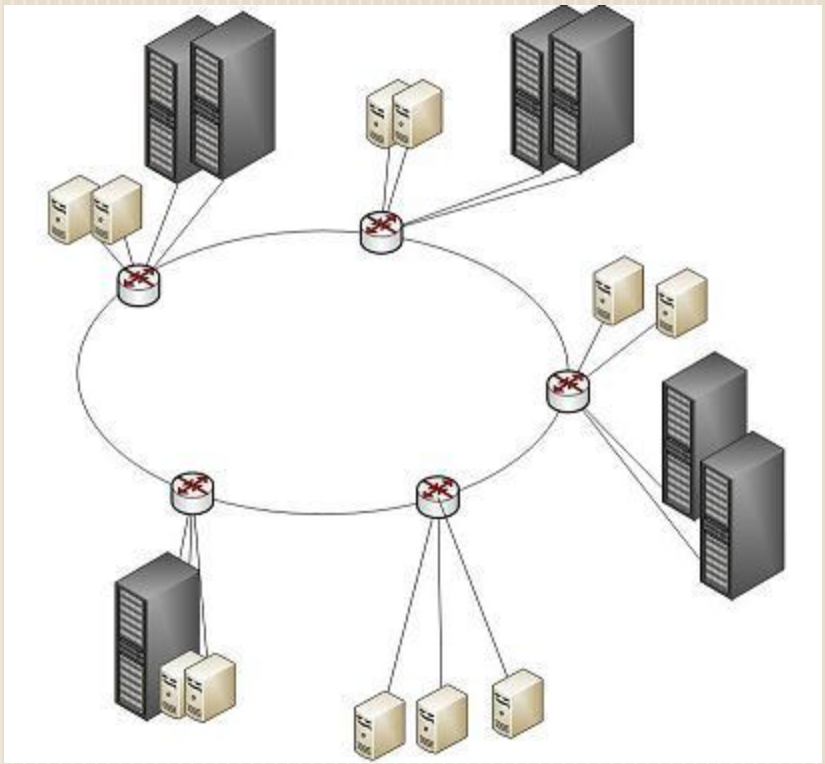
SAN

- Используются протоколы SCSI и SAS, iSCSI (tcp транспорт), FCP (Fiber Channel Protocol)

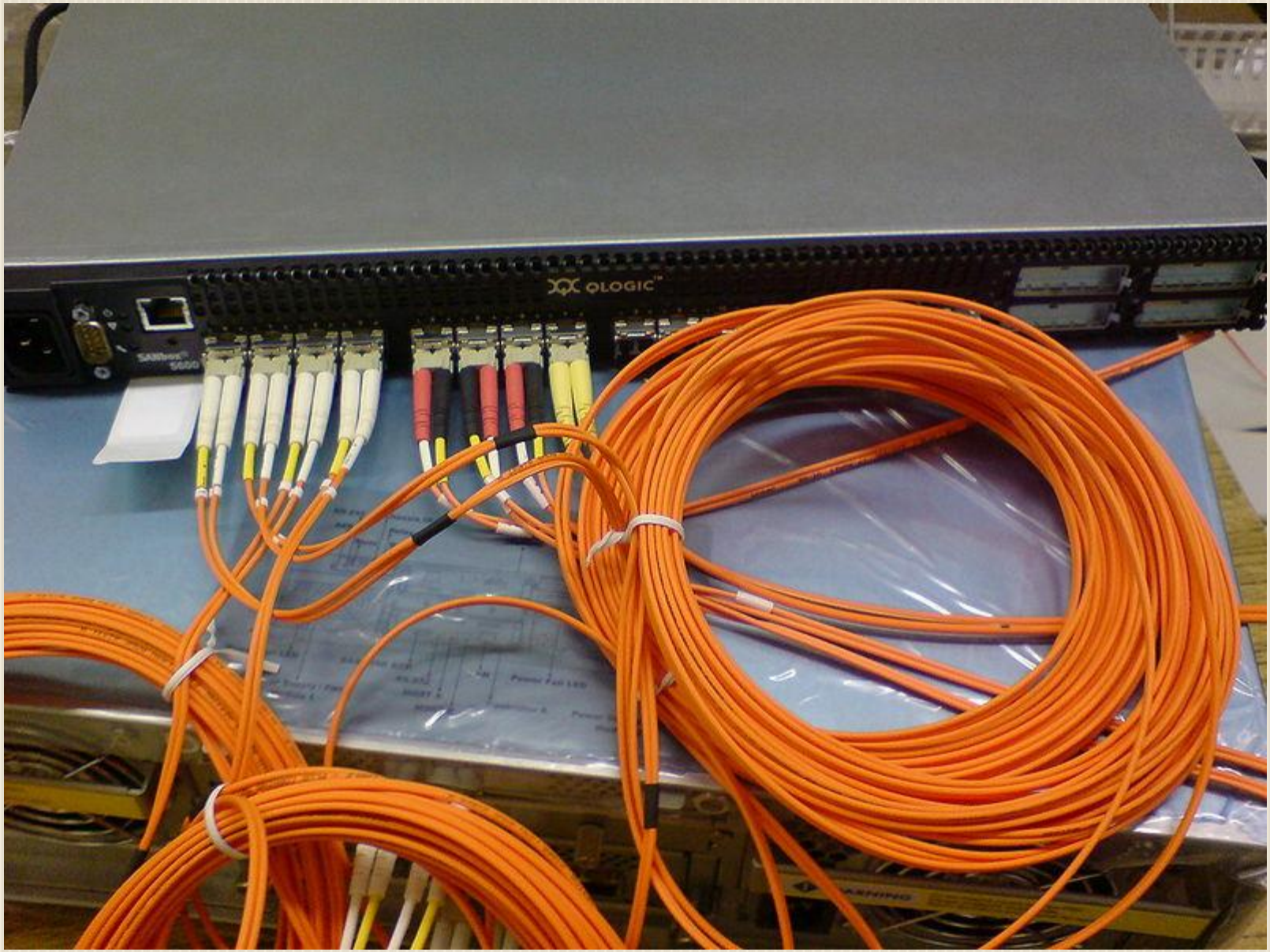


SAN



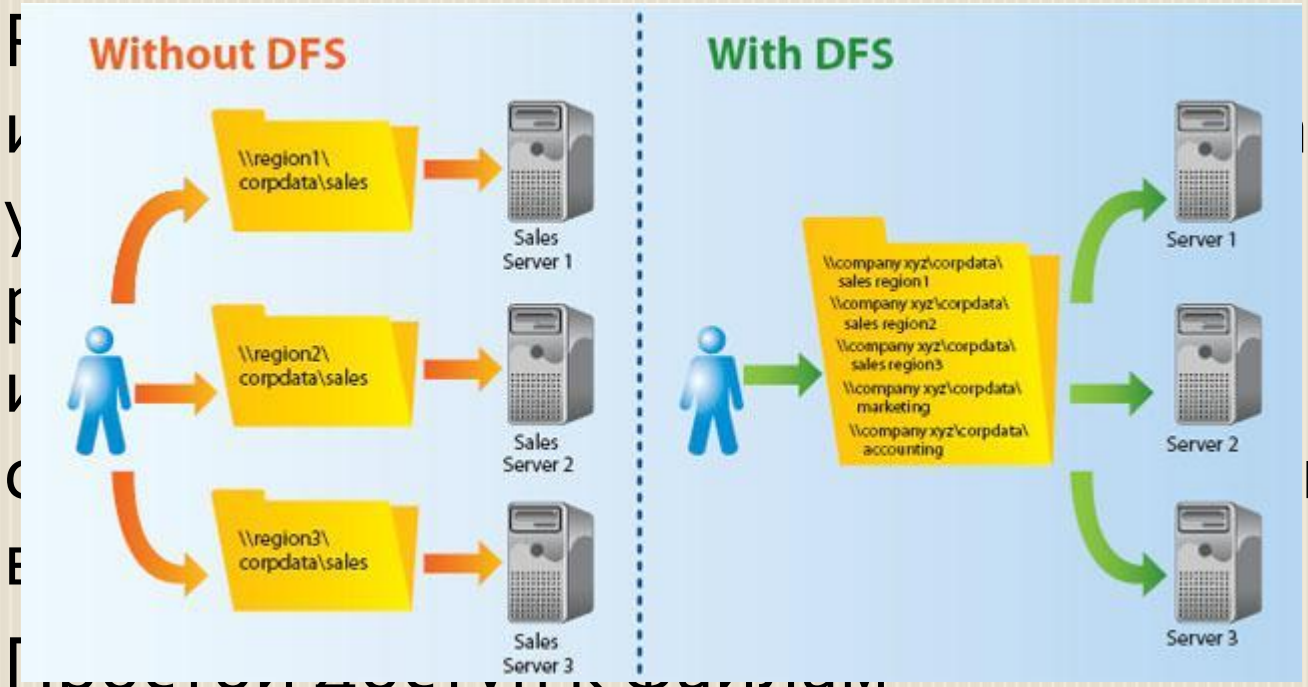


Switch SAN



DFS (**Distributed File System**)

- Р
  и
  у
  р
  и
  с
  в



и

ые по

мися

- Прозрачный доступ к файлам
- Доступность
- Эффективная загрузка сервера
- Безопасность файлов и папок

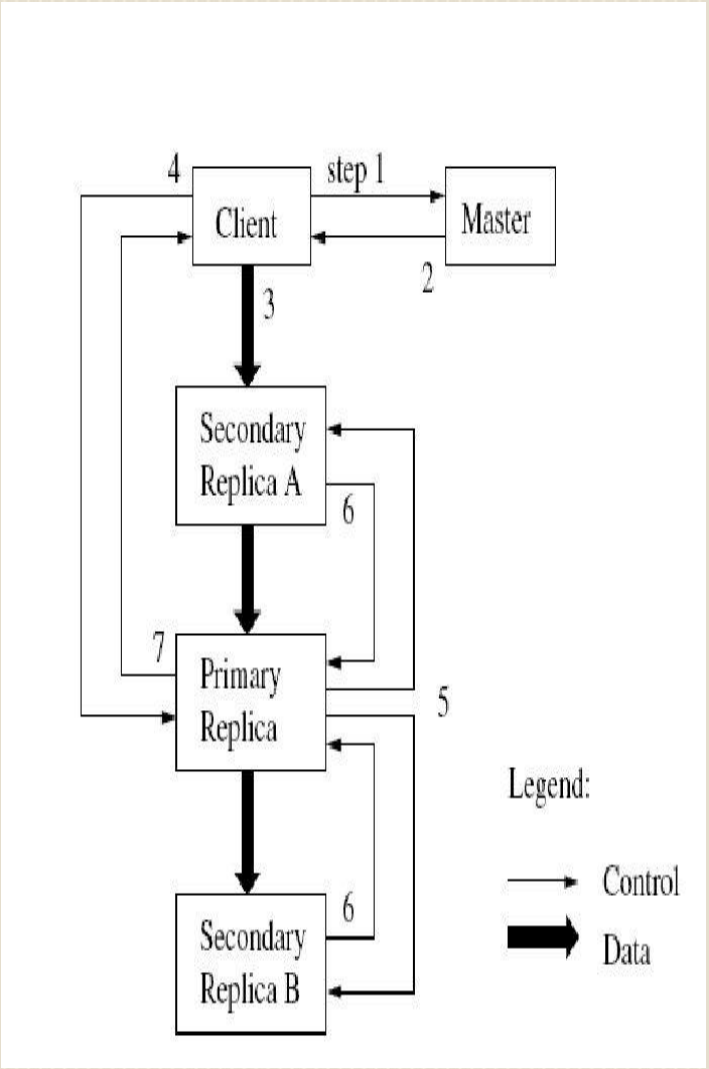
GFS (Google File System)

- Система строится из большого количества обыкновенного недорого оборудования, которое часто дает сбои. Должны существовать мониторинг сбоев, и возможность в случае отказа какого-либо оборудования восстановить функционирование системы.
- Система должна хранить много больших файлов. Как правило, несколько миллионов файлов, каждый от 100 Мб и больше. Также часто приходится иметь дело с многогигабайтными файлами, которые также должны эффективно храниться. Маленькие файлы тоже должны храниться, но для них не оптимизируется работа системы.
- Как правило, встречаются два вида чтения: чтение большого последовательного фрагмента данных и чтение маленького объема произвольных данных. При чтении большого потока данных обычным делом является запрос фрагмента размером в 1Мб и больше. Такие последовательные операции от одного клиента часто читают подряд идущие куски одного и того же файла. Чтение небольшого размера данных, как правило, имеет объем в несколько килобайт. Приложения, критические по времени исполнения, должны накопить определенное количество таких запросов и отсортировать их по смещению от начала файла. Это позволит избежать при чтении блужданий вида назад-вперед.
- Часто встречаются операции записи большого последовательного куска данных, который необходимо дописать в файл. Обычно, объемы данных для записи такого же порядка, что и для чтения. Записи небольших объемов, но в произвольные места файла, как правило, выполняются не эффективно.
- Система должна реализовывать строго очерченную семантику параллельной работы нескольких клиентов, в случае если они одновременно пытаются дописать данные в один и тот же файл. При этом может случиться так, что поступят одновременно сотни запросов на запись в один файл. Для того чтобы справится с этим, используется атомарность операций добавления данных в файл, с некоторой синхронизацией. То есть если поступит операция на чтение, то она будет выполняться, либо до очередной операции записи, либо после.
- Высокая пропускная способность является более предпочтительной, чем маленькая задержка. Так, большинство приложений в Google отдают предпочтение работе с большими объемами данных, на высокой скорости, а выполнение отдельно взятой операции чтения и записи, вообще говоря, может быть растянуто.

GFS устройство

- Chunk (Чанк сервера) – сервера с самостоятельной операционной системой сохраняющие данные на установленные локально физические устройства.
- Master (мастер сервер) – хранит три важных вида метаданных: пространства имен файлов и чанков, отображение файла в чанки и положение реплик чанков. Все метаданные хранятся в памяти мастера.
- Он выполняется сканирование чанк-серверов в фоновом режиме. Эти периодические сканирования используются для сборки мусора, дополнительных репликаций, в случае обнаружения недоступного чанк-сервера и перемещение чанков, для балансировки нагрузки и свободного места на жестких дисках чанк-серверов.
- Мастер отслеживает положение чанков. При старте чанк-сервера мастер запоминает его чанки. В процессе работы мастер контролирует все перемещения чанков и состояния чанк-серверов. Таким образом, он обладает всей информацией о положении каждого чанка.
- Важная часть метаданных — это лог операций. Мастер хранит последовательность операций критических изменений метаданных. По этим отметкам в логе операций, определяется логическое время системы. Именно это логическое время определяет версии файлов и чанков.
- Так как лог операций важная часть, то он должен надежно храниться, и все изменения в нем должны становиться видимыми для клиентов, только когда изменятся метаданные. Лог операций реплицируется на несколько удаленных машин, и система реагирует на клиентскую операцию, только после сохранения этого лога на диск мастера и диски удаленных машин.
- Мастер восстанавливает состояние системы, исполняя лог операций. Лог операций сохраняет относительно небольшой размер, сохраняя только последние операции. В процессе работы мастер создает контрольные точки, когда размер лога превосходит некоторой величины, и восстановить систему можно только до ближайшей контрольной точки. Далее по логу можно заново воспроизвести некоторые операции, таким образом, система может откатываться до точки, которая находится между последней контрольной точкой и текущем временем.

Схема



- Клиент спрашивает мастера, какой из чанк-серверов владеет чанком, и где находится этот чанк в других репликах. Если необходимо, то мастер отдает чанк кому-то во владение.
- Мастер в ответ выдает первичную реплику, и остальные (вторичные) реплики. Клиент хранит эти данные для дальнейших действий. Теперь, общение с мастером клиенту может понадобиться только, если первичная реплика станет недоступной.
- Далее клиент отсылает данные во все реплики. Он может это делать в произвольном порядке. Каждый чанк-сервер будет их хранить в специальном буфере, пока они не понадобятся или не устареют.
- Когда все реплики примут эти данные, клиент посылает запрос на запись первичной реплике. В этом запросе содержатся идентификация данных, которые были посланы в шаге 3. Теперь первичная реплика устанавливает порядок, в котором должны выполняться все изменения, которые она получила, возможно от нескольких клиентов параллельно. И затем, выполняет эти изменения локально в этом определенном порядке.
- Первичная реплика пересылает запрос на запись всем вторичным репликам. Каждая вторичная реплика выполняет эти изменения в порядке, определенном первичной репликой.
- Вторичные реплики рапортуют об успешном выполнении этих операций.
- Первичная реплика шлет ответ клиенту. Любые ошибки, возникшие в какой-либо реплике, также отсылаются клиенту. Если ошибка возникла при записи в первичной реплике, то и запись во вторичные реплики не происходит, иначе запись произошла в первичной реплике, и подмножестве вторичных. В этом случае клиент обрабатывает ошибку и решает, что ему дальше с ней делать.

Устойчивость к сбоям и диагностика ошибок

- Мастер всегда записывает данные в лог и передает его на несколько копий мастеров (Shadow Master) только после успешной записи на все сервера производится запись в чанки
- Каждый чанк-сервер должен самостоятельно определять целостность данных.
  Каждый чанк разбивается на блоки длиной **64 Кбайт**. Каждому такому блоку соответствует **32**-битная контрольная сумма. Как и другие метаданные эти суммы хранятся в памяти, регулярно сохраняются в лог, отдельно от данных пользователя.
  Перед тем как считать данные чанк-сервер проверяет контрольные суммы блоков чанка, которые пересекаются с затребованными данными пользователем или другим чанк-сервером. То есть чанк-сервер не распространяет испорченные данные. В случае несовпадения контрольных сумм, чанк-сервер возвращает ошибку машине, подавшей запрос, и рапортует о ней мастеру. Пользователь может считать данные из другой реплики, а мастер создает еще одну копию из данных другой реплики. После этого мастер дает инструкцию этому чанк-серверу об удалении этой испорченной реплики.