Бойцов Артём Игоревич

Практическая работа № 3.3. Частные вопросы управления ИТ-инфраструктурой: примеры инфраструктурных решений, применяющихся в крупных сетевых проектах

Пример реализации инфраструктуры в Google

Google — это огромная сервисно-ориентированная платформа для построения масштабируемых приложений, позволяющая выпускать и поддерживать множество конкурентоспособных интернет-приложений, работающих на уровне всей глобальной сети. Компания ставит перед собой цель постоянно строить все более и более производительную и масштабируемую инфраструктуру для поддержки своих продуктов.

GFS является наиболее, наверное, известной распределенной файловой системой. Надежное масштабируемое хранение данных крайне необходимо для любого приложения, работающего с таким большим массивом данных, как все документы в интернете. GFS является основной платформой хранения информации в Google. GFS — большая распределенная файловая система, способная хранить и обрабатывать огромные объемы информации.

В системе существуют мастер-сервера и чанк-сервера, собственно, хранящие данные. Как правило, GFS-кластер состоит из одной главной машины мастера (master) и множества машин, хранящих фрагменты файлов чанк-серверы (chunkservers). Клиенты имеют доступ ко всем этим машинам. Файлы в GFS разбиваются на куски — чанки (chunk, можно сказать фрагмент). Чанк имеет фиксированный размер, который может настраиваться. Каждый такой чанк имеет уникальный и глобальный 64 — битный ключ, который выдается мастером при создании чанка. Чанк-серверы хранят чанки, как обычные Linux файлы, на локальном жестком диске. Для надежности каждый чанк может реплицироваться на другие чанк-серверы. Обычно используются три реплики.

Каждое изменение чанка должно дублироваться на всех репликах и изменять метаданные. В GFS мастер дает чанк во владение (lease) одному из серверов, хранящих этот чанк. Такой сервер называется первичной (primary) репликой. Остальные реплики объявляются вторичными (secondary). Первичная реплика собирает последовательные изменения чанка, и все реплики следуют этой последовательности, когда эти изменения происходят. Механизм владения чанком устроен таким образом, чтобы минимизировать нагрузку на мастера. При выделении памяти сначала выжидается 60 секунд. А затем, если потребуется первичная реплика может запросить мастера на расширение этого интервала и, как правило, получает положительный ответ. В течение этого выжидаемого периода мастер может отменить изменения.

Мастер важное звено в системе. Он управляет репликациями чанков: принимает решения о размещении, создает новые чанки, а также координирует различную деятельность внутри системы для сохранения чанков полностью реплицированными, балансировки нагрузки на чанксерверы и сборки неиспользуемых ресурсов. В отличие от большинства файловых систем GFS не хранит состав файлов в директории. GFS логически представляет пространство имен, как таблицу, которая отображает каждый путь в метаданные.

Авторы системы считают одной из наиболее сложных проблем частые сбои работы компонентов системы. Количество и качество компонентов делают эти сбои не просто исключением, а скорее нормой. Сбой компонента может быть вызван недоступностью

этого компонента или, что хуже, наличием испорченных данных. GFS поддерживает систему в рабочем виде при помощи двух простых стратегий: быстрое восстановление и репликации. Быстрое восстановление — это, фактически, перезагрузка машины. При этом время запуска очень маленькое, что приводит к маленькой заминке, а затем работа продолжается штатно. При репликации мастер реплицирует чанк, если одна из реплик стала недоступной, либо повредились данные, содержащие реплику чанка. Поврежденные чанки определяется при помощи вычисления контрольных сумм. Еще один вид репликаций в системе — это репликация мастера. Реплицируется лог операций и контрольные точки (checkpoints).

MapReduce является программной моделью и соответствующей реализацией обработки и генерации больших наборов данных. Пользователи могут задавать функцию, обрабатывающую пары ключ/значение для генерации промежуточных аналогичных пар, и сокращающую функцию, которая объединяет все промежуточные значения, соответствующие одному и тому же ключу. Многие реальные задачи могут быть выражены с помощью этой модели. Программы, написанные в таком функциональном стиле, автоматически распараллеливаются и адаптируются для выполнения на обширных кластерах. Система берет на себя детали разбиения входных данных на части, составления расписания выполнения программ на различных компьютерах, управления ошибками, и организации необходимой коммуникации между компьютерами.

BigTable является крупномасштабной, устойчивой к потенциальным ошибкам, самоуправляемой системой, которая может включать в себя терабайты памяти и петабайты данных, а также управлять миллионами операций чтения и записи в секунду. BigTable представляет собой распределенный механизм хэширования, построенный поверх GFS, а вовсе не реляционную базу данных и, как следствие, не поддерживает SQL-запросы и операции типа Join. Она предоставляет механизм просмотра данных для получения доступа к структурированным данным по имеющемуся ключу.

Пример реализации инфраструктуры для проекта Flickr

Flickr является мировым лидером среди сайтов размещения фотографий. Перед Flickr стоит крайне непростая задача, они должны контролировать огромное количество ежесекундно обновляющегося контента, непрерывно пополняющиеся пользователи, постоянный поток новых предоставляемых пользователям возможностей, и при этом поддерживать постоянно высокий уровень производительности.

Входные запросы поступают на сдублированные контроллеры приложений Brocade ServerIron ADX. Они обеспечивают коммутацию приложений и балансировку трафика. Коммутатор приложений осуществляет трансляцию адресов после выбора нужного сервера, причем сами адреса серверов скрыты.

В основе масштабируемости лежит репликация. Для поиска по определенной части базы данных создается отдельная копия этого фрагмента. Активная репликация производится по принципу мастер-мастер. Автоматическое инкрементирование идентификационных номеров используется для поддержания системы в режиме одновременной активности обоих серверов в паре. При этом привязывание новых учетных записей к сегментам системы происходит случайным образом. Миграция пользователей проводится время от времени для того, чтобы избавиться от проблем, связанных с излишне активными пользователями.

Каждый сервер в рамках одного сегмента в обычном состоянии нагружен ровно на половину. При выключении половины серверов в каждом сегменте система продолжит функционировать без изменений.

Организация резервного копирования данных реализована с помощью процесса ibbackup, который выполняется регулярно посредством cron daemon'a, причем на каждом сегменте он настроен на разное время. Каждую ночь делается снимок со всего кластера баз данных.