Устройство Object Storage (LeoFS)

Александр Чистяков

highload++ 2014

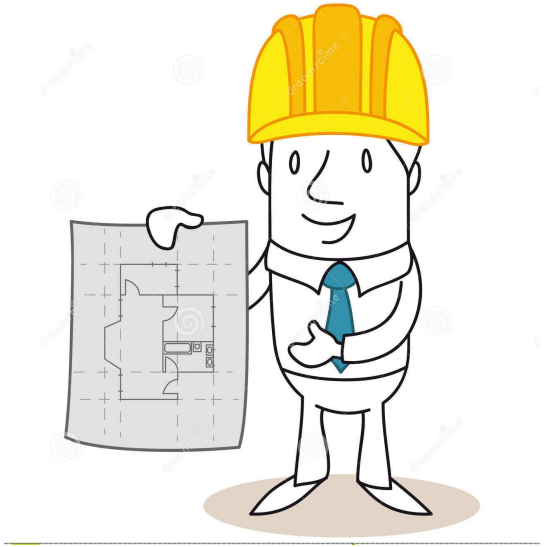
Конференция разработчиков
высоконагруженных систем

Разрешите представиться

- Меня зовут Саша
- Я работаю системным
- Главным
- В прошлом — резидент района «Гражданка»
- Сейчас я вам расскажу одну историю

Позвольте узнать

- Почему вы не на докладе Димы Смирнова?
- Вы делаете хайлоуд веб-проекты?
- Программисты?
- Инженеры отделов эксплуатации?
- Архитекторы?
- Прогуливаете работу?



Содержание пред. серий

- HighLoad 2012 — https://clck.ru/9Lutd
- DevConf 2014 — https://clck.ru/9Lutw
- Setup.ru — конструктор сайтов
- Много (десятки миллионов) пользовательских файлов
- Метаинформация хранится в БД
- Сами файлы тоже хранятся в БД

  (на самом деле — нет, LO API)

Новейшая история

- Проект на языке Perl
- Разрабатывается отличной командой

  удаленщиков под руководством Дмитрия Симонова
- Все большие проблемы решены, кроме одной:
- **На серверах у выбранного хостера (Hetzner) не хватает места для хранения пользовательского контента**
- А пользователи не хотят переставать его генерировать

Варианты решения

- Купить аккаунт в Amazon S3
- Не покупать аккаунт в Amazon S3
  - Сделать Amazon S3 самим!
  - Это называется «Object Storage»:
  - Elliptics (Yandex)
  - OpenStack Swift
  - Ceph Object Gateway
  - Riak CS
  - LeoFS

Общие свойства

- HTTP REST или S3 интерфейс
- Автоматическая репликация
- Автоматический failover
- Относительно легкое добавление новых узлов в хранилище
- Тем не менее, надо сделать выбор!

Устройство object storage

- Узлы, на которых хранятся сами данные (имеют локальное хранилище)
- Узлы, на которых хранится метаинформация (могут совпадать с первыми)
- Узлы, занимающиеся маршрутизацией запросов к первым двум типам узлов
- Узлы, запускающие и контролирующие фоновые задачи
- Таким образом, любой object storage — это **локальные хранилища + маршрутизатор запросов к ним**

Технические соображения

- Erlang — как Python, но лучше*
  (LeoFS и Riak CS против Swift)
- Mnesia и LevelDB лучше*, чем SQLite
  опять (LeoFS и Riak CS против Swift)
- DuckDuckGo лучше*, чем Yandex
  (что угодно против Elliptics)
- Встроенное в систему кэширование
  лучше*, чем внешнее (LeoFS против всех)

* утверждение может содержать ложь

Кроме того

- LeoFS сделана в Японии
  - Японская сталь
  - Японские ножи
  - Японские автомобили
  - Язык Ruby
  - Аниме
  - Айкидо
- Надо брать!

Выбор технологии

- Выбор технологии — это прыжок веры
- Эффект Даннинга-Крюгера
- Ошибка выживших
- И другие когнитивные искажения
- «Главная проблема цитат в интернете
  в том, что люди сразу верят
  в их подлинность»

  В.И.Ленин

Устройство LeoFS

- LeoFS manager — хранит метаинформацию (на какой узел хранилища за каким файлом идти)
- LeoFS storage — хранит сами файлы
- LeoFS gateway — транслирует HTTP запросы

  в запросы к узлам manager и storage,

  кэширует контент локально
- Чтобы LeoFS manager не являлся SPOF,

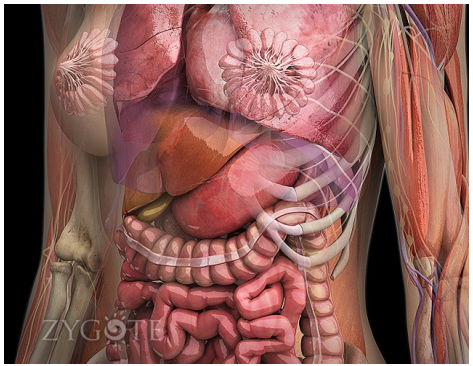
  их нужно два: master и slave

Устройство LeoFS

- У LeoFS manager в качестве хранилища выступает Mnesia
- У LeoFS storage в качестве хранилища выступает LevelDB
- Пара слов про Mnesia:
  - Распределенная СУБД
  - Входит в стандартную поставку языка Erlang
- Пара слов про LevelDB:
  - Key-value storage
  - Разработана в Google
  - Представляет собой LSM-tree

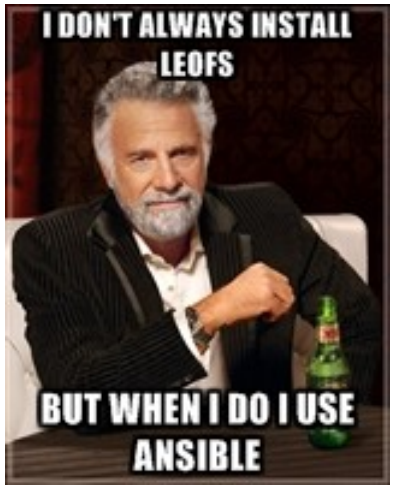
Устройство LeoFS

- Внутренние процессы:
- LevelDB — нужно делать compaction
- Маршрутизация — нужно делать перестроение кольца (добавил, *удолил*)
- Временно выключить/включить узел
- Все эти процессы запускаются **вручную**
- Лично я ненавижу автоматику!



Развертывание LeoFS

- Когда мне нужно что-нибудь развернуть, я пишу Ansible playbooks:
- https://clck.ru/9Lx73
- Определите host variables и запустите Ansible — все должно развернуться автоматически



Отказоустойчивость

- В конфигурационном файле задается:
  - Количество реплик
  - Количество успешных операций
    - Чтения
    - Записи
    - Удаления

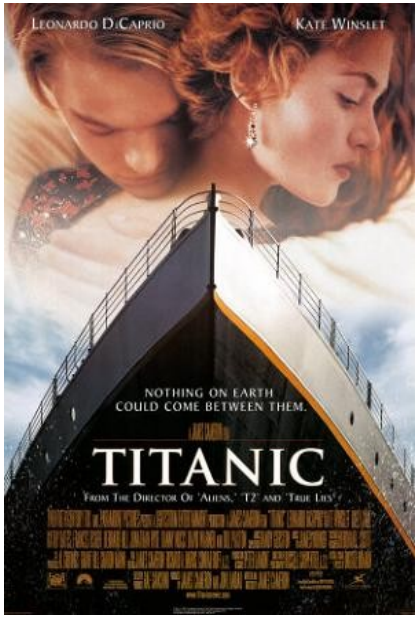
100% HA

- Как любовь с первого взгляда:

  с кем-то случается, а я вот к своему счастью

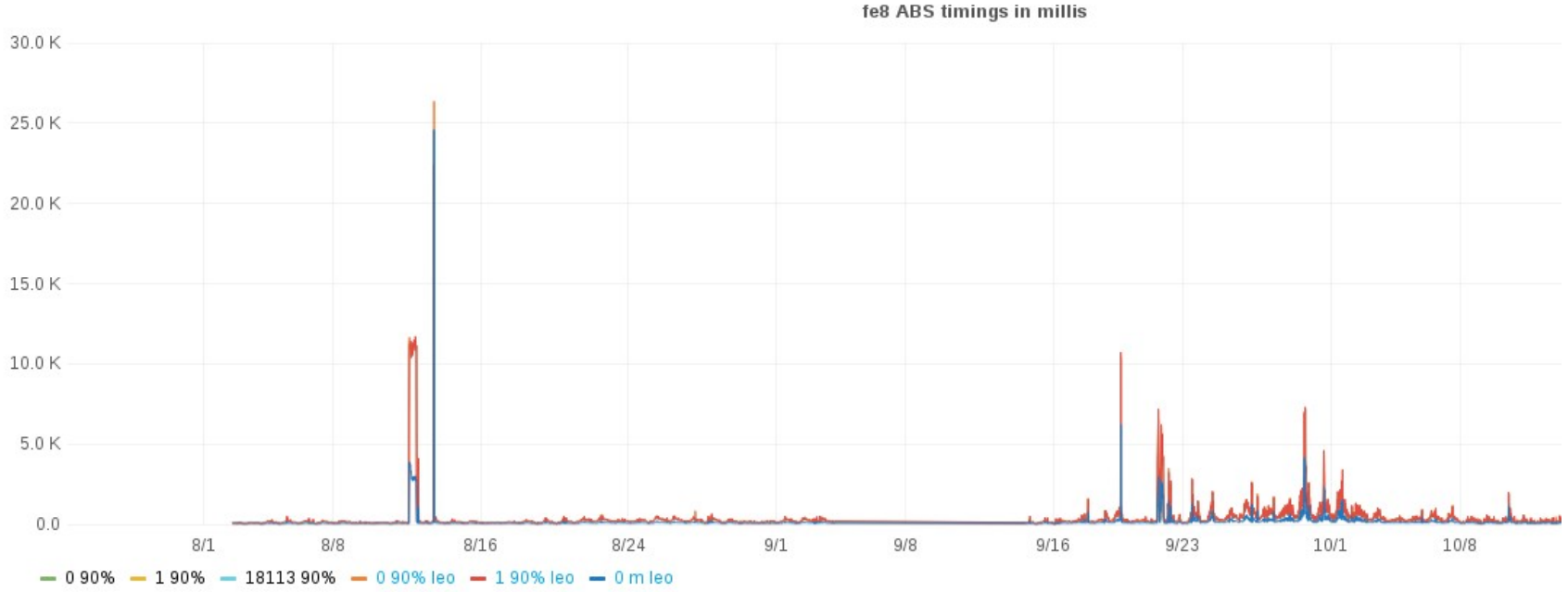
  три месяца привыкал
- Пока — 100% HA даже у меня
- (Уже наступившие события имеют

  вероятность 100%)



Как это было

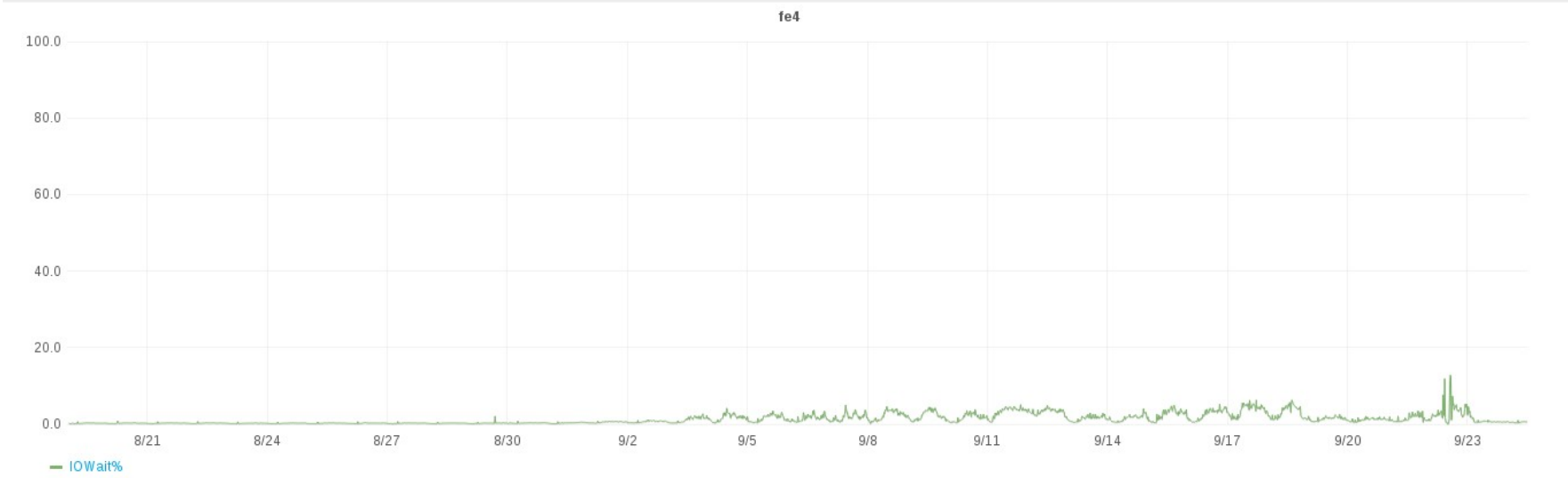
- Найдите на графике момент отказа



Нагрузка на сеть

Нагрузка на диск

- Не заслуживает упоминания

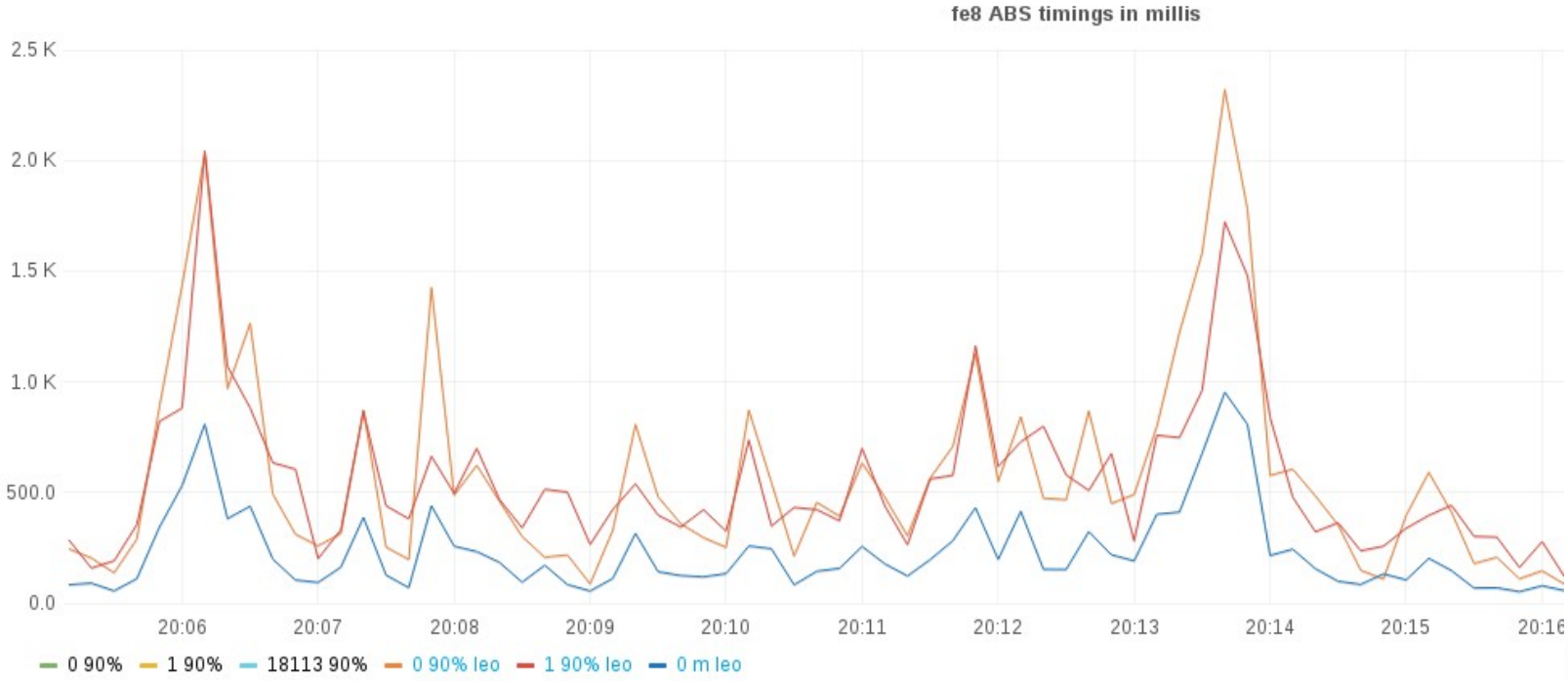


Поговорим о хорошем

- https://clck.ru/9LwvD - муки выбора технологии (12 июня)
- Тогда же — начало внедрения, первые коммиты в репозиторий
- К августу все было развернуто в продакшн
- В сентябре миграция была полностью завершена прозрачно для пользователей
- В августе у нас умер один из storage узлов (последовательный отказ двух дисков) — мы просто заменили его на новый пустой

Поговорим о ~~хорошем~~

- График latency говорит сам за себя (время в ms):



Поговорим о разном

- Клиентские библиотеки S3 не очень высокого качества — многие библиотеки вообще не понимают, что S3-хранилище может быть не по адресу s3.amazonaws.com
- В случае отказа узла latency становится еще выше
- Граничные условия — у нас никогда не происходит модификация контента (для S3-like хранилищ это оптимально)
- Балансировка контента производится не автоматически (и это прекрасно!)

Граничные условия

- Статический контент
- Никогда не модифицируется (новый контент — новый URL)
- Никогда не удаляется (зачем?)
- Пользователи согласны подождать (не видео раздаем)

Выводы

- *Некоторые* object storages вполне можно использовать в бою
- Некоторые — наверное, нельзя
- Отличить первые от вторых можно
  а) опытным путем,
  б) через откровение свыше
- См. «методика проведения физического эксперимента»

Спасибо за внимание!

- Пожалуйста, ваши вопросы?
- С вами был Саша Чистяков
- Главный инженер компании Git in Sky
- Координатор SPb. DevOps Meetup
- alex@gitinsky.com
- http://twitter.com/noatbaksap
- http://slideshare.com/alexclear
- **Спасибо Олегу Цареву за архиватор и кров**