Решардинг Redis без даунтайма

Роман Павлушко
AVITO.ru

DevConf

AVITO.RU
ЛЕГКО КУПИТЬ, ЛЕГКО ПРОДАТЬ

Самый большой сайт бесплатных объявлений

Частные объявления и объявления компаний о продаже. Домашние животные, бытовая техника, электроника, недвижимость, автомобили, одежда и многое другое. На сайте **6 835 422 объявления**

ПОДАТЬ ОБЪЯВЛЕНИЕ

Регистрация | Вход

В лето с призами!

Участвовать в конкурсе

Москва
Санкт-Петербург
Екатеринбург
Казань
Краснодар

Астраханская обл.
Башкортостан
Белгородская обл.
Брянская обл.
Владимирская обл.
Волгоградская обл.
Вологодская обл.
Воронежская обл.
Ивановская обл.
Калининградская обл.
Калужская обл.
Кировская обл.
Краснодарский край
Курганская обл.
Курская обл.
Ленинградская обл.
Липецкая обл.
Марий Эл
Московская обл.
Мурманская обл.
Нижегородская обл.

Нижний Новгород
Ростов-на-Дону
Самара
Уфа
Челябинск

Новгородская обл.
Оренбургская обл.
Пензенская обл.
Пермский край
Псковская обл.
Ростовская обл.
Рязанская обл.
Самарская обл.
Саратовская обл.
Свердловская обл.
Смоленская обл.
Ставропольский край
Тамбовская обл.
Татарстан
Тверская обл.
Тульская обл.
Удмуртия
Ульяновская обл.
Челябинская обл.
Чувашия
Ярославская обл.

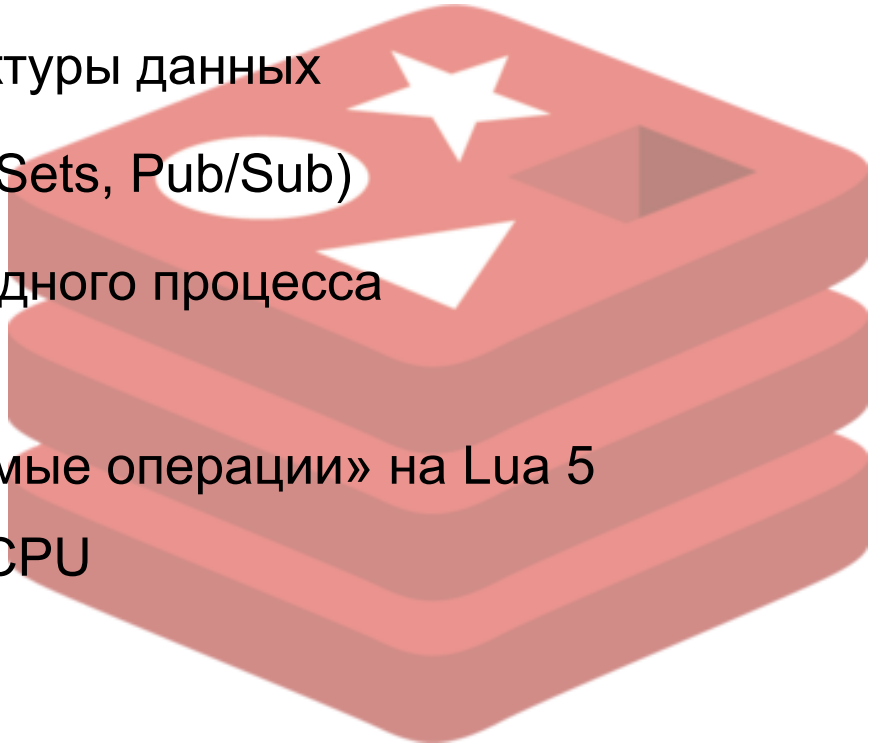
О проекте

- В TOP10 сайтов Рунета
- 90M+ запросов в сутки (AVITO, TORG, API)
- 130K+ запросов в минуту
- 230K+ новых объявлений в сутки
- 400K+ картинок в день
- 8M+ пользователей размещали объявления
- 50M новых объявлений за последний год

О Redis

- Хранит и работает с данными только из памяти
- На диск пишет только для сохранности
- Поддерживает различные структуры данных

  (Strings, Hash, List, Sets, Sorted Sets, Pub/Sub)
- Подобие транзакций в рамках одного процесса
- Репликация из коробки
- С версии 2.6 появились «хранимые операции» на Lua 5
- 1 инстанс использует > 1 ядра CPU



Клиентский протокол Redis

```
*<number of arguments> CR LF
$<number of bytes of argument 1> CR LF
<argument data> CR LF
...
$<number of bytes of argument N> CR LF
<argument data> CR LF
```

Подробнее http://redis.io/topics/protocol

Протокол репликации Redis

1. Слейв посылает команду SYNC мастеру
2. Мастер стартует синхронизацию rdb-снапшота + запускает лог запросов
3. По завершении SYNC мастер отправляет rdb-снапшот слейву
4. Слейв вычитывает rdb-снапшот и начинает принимать команды от мастера

Подробнее о репликации http://redis.io/topics/replication

Подробнее о формате rdb http://clck.ru/1AyyU

Почему мы используем Redis?

- 400M постоянно обновляемых документов
- 3K+ уникальных запросов write/read в секунду (8K пик)
- Данные должны одинаково быстро отдаваться на всем протяжении их жизни
- Целостностью в маленьких масштабах можно пренебречь
- Коробочное решение, поставил и забыл (почти правда)
- Прост в развертывании и администрировании

Для чего мы используем Redis

- Счетчики для статистики и мониторинга
- Закладки пользователей
- Очереди сообщений

Проблемы роста данных

2009 год:

- 100К запросов/сутки
- 5 нод с заделом на будущее

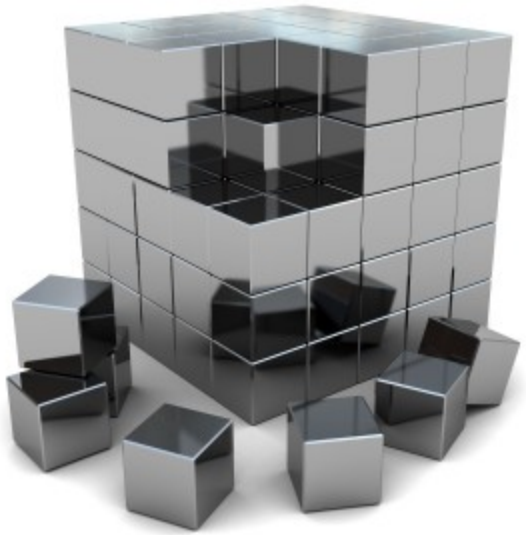
2012 год:

- 20М запросов/сутки
- 250М документов
- 35ГБ данных

Проблема одна — медленная синхронизация данных на диск

Решение медленной синхронизации

- Разбиваем ноды на более мелкие
- Поднимаем несколько нод на 1 сервере
- Получаем равномерную нагрузку на диск



Задачи и требования

1. Распределить данные по новым нодам
2. Изменить алгоритм распределения ключей
3. Вычистить старые версии данных
4. Не потерять данные
5. Не просидеть за этой задачей «год»

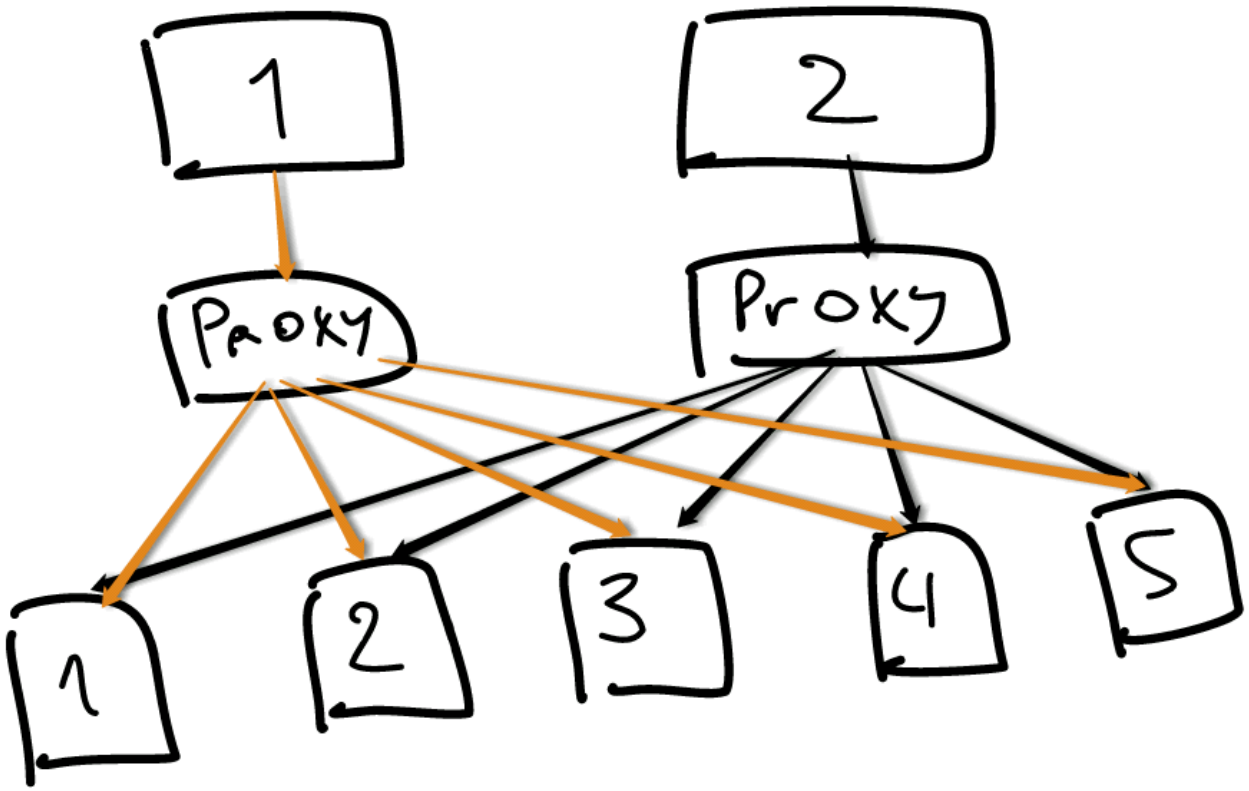
Ищем решение

- Переместить данные со старых на новые ноды скриптом
  - нужен длительный downtime
  - нужно получить список ключей (keys *)
- Прослойка в протокол репликации (как делал Skype)

  http://www.slideshare.net/profyclub_ru/redis-9518928
  - нельзя перераспределить данные по нодам
  - необходима небольшая остановка на запись в Redis

    (в нашем случае много подготовительной работы)

Что нам надо



Что включает *Proxy*

- Надежный сетевой демон
- Протокол репликации Redis
  - Парсер формата rdb
  - Парсер протокола Redis
- Алгоритм распределения ключей
- Фильтры устаревших данных

В Redis уже все есть, почти все...

- Надежный сетевой демон
- Протокол репликации Redis
  - Парсер формата rdb
  - Парсер протокола Redis
- Алгоритм распределения ключей
- Фильтры устаревших данных

Форкаем Redis и вбиваем в него костыли

1. Добавляем в код redis-клиентов для новых нод
2. В разбор rdb добавляем функцию, транслирующую документы в команды
3. Добавляем фильтр на старые версии данных
4. Заменяем проигрывание команд собственной функцией
5. Собственная функция игнорирует данные или отправляет их на новые ноды

Процесс решардинга ноды

1. Разворачиваем инстансы для нового кластера
2. Настраиваем мониторинг и резервное копирование
3. Поднимаем под каждой старой нодой по Proxy-инстансу
4. Последовательно стартуем Proxy-инстансы, чтобы не забить сеть rdb-снапшотами
5. Когда Proxy всех нод перешел в режим проигрывания комманд с мастера, значит уже можно переключать приложение на новый кластер

Результаты

Разработка 2 недели

В фоновом режиме в паре с системным администратором.

Процесс решардинга 6 часов

Patch

1. **replication.c** : int connectWithMaster(void)
2. **redis.c** : int processCommand(redisClient *c)
3. **rdb.c** : int rdbLoad(char *filename)
4. **resharding.c** *

Исходники https://github.com/prn/redis-resharding

Diff http://clck.ru/1AzbW

WARNING!

1. Избегайте тяжелых запросов, Redis не для этого
2. Используйте максимум 1 инстанс на ядро, а лучше 0,75 :)
3. Несколько коллекций (баз) синхронизируется на диск в общий файл
4. Тщательно тестируйте новые версии Redis перед обновлением, особенно если версии RDB меняются
5. Не храните очень ценные данные в Redis

Ищем разработчиков

HTML/JavaScript, PHP, PostgreSQL, Sphinx, Unix

Если вы считаете, что вы эксперт в какой-нибудь из перечисленных технологий, отправляйте резюме!

http://www.avito.ru/job

Спасибо!
Вопросы?

Роман Павлушко

rpavlushko@avito.ru

twitter.com/pavlushko

slideshare.net/pavlushko/